Design of a High Speed Data Capture Device for a Coherent Radar Application

Mark Frankford and Michael A. Carr

The Ohio State University

ElectroScience Laboratory

Department of Electrical Engineering
Columbus, Ohio 43212

Final Report 746389-1
Grant No. N00173-04-2-C005

November 2006

Attn: James J. Genova, Code 5760
Naval Research Laboratory
4555 Overlook Ave., SW
Washington, D.C. 20375

REPORT DOCUMENTATION PAGE	1. REPORT NO.	2.	3. Recipient's Accession No.
4. Title and Subtitle Design of a High Speed Data Capture Device for a Coherent Radar Application			**5. Report Date** November 2006 **6.**
7. Author(s) Mark Frankford and Michael A. Carr			**8. Performing Org. Rept. No.** 746389-1
9. Performing Organization Name and Address **The Ohio State University** **ElectroScience Laboratory** **1320 Kinnear Road** **Columbus, OH 43212**			**10. Project/Task/Work Unit No.** **11. Contract I or Grant (G) No.** I (G) N00173-04-2-C005
12. Sponsoring Organization Name and Address Naval Research Laboratory 4555 Overlook Ave., SW Washington, D.C. 20375			**13. Report Type/Period Covered** Final Report **14.**
15. Supplementary Notes			

16. Abstract (Limit: 200 words)

Anti-ship missiles (ASM) have long presented a serious threat to the safety and security of America's naval forces. Over the past 30 years, significant efforts have been made to develop reliable countermeasures to protect the fleet against a wide variety of ASM weaponry. Due to cost, weight, and size limitations, conventional radar-guided ASMs (RGASM) have employed non-coherent radar techniques, and thus countermeasures developed to date have been designed specifically to defeat non-coherent threats. Recently advances in miniaturization have enabled the design of coherent RGASMs, demanding the creation of a new breed of countermeasures.

To enable the design of countermeasures to protect against coherent RGASMs, a variety of tools must first be constructed. Amount these is a coherent RGASM test bed to be used for monitoring the behavior of missiles as they are deployed against simulated targets in a laboratory environment. One component of this test bed consists of a high speed data capture device (HSCD) for capturing and recording real-time data as it moves through a RGASM's digital processing in order to analyze how the RGASM's decision making is affected by each countermeasure's behavior. This thesis outlines the design of an HSCD for interfacing directly to a RGASM to aid in accomplishing this task.

17. Document Analysis

a. Descriptors

HIGH-SPEED DATA CAPTURE; ANTI-SHIP MISSILE; COHERENT RADAR; REAL-TIME; SIMULATION; COUNTERMEASURES

b. Identifiers/Open-Ended Terms

c. COSATI Field/Group

18. Availability Statement	19. Security Class (This report)	21. No. of Pages
Approval for release is required; Distribution is limited.	Unclassified (ITAR)	91
	20. Security Class (This page) Unclassified	**22. Price**

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Motivation

Anti-ship missiles (ASM) have long presented a serious threat to the safety and security of America's naval forces. Over the past 30 years, significant efforts have been made to develop reliable countermeasures to protect the fleet against a wide variety of ASM weaponry. Due to cost, weight, and size limitations, conventional radar-guided ASMs (RGASM) have employed non-coherent radar techniques, and thus countermeasures developed to date have been designed specifically to defeat non-coherent threats. Recently, advances in miniaturization have enabled the design of coherent RGASMs, demanding the creation of a new breed of countermeasures.

Two sets of data are required to properly gauge the effectiveness of a given countermeasure against a coherent RGASM tracking algorithm. The first set of data consists of the raw signals from the radar's receiver, while the second set of data consists of the decisions made by the tracking algorithm. By comparing these two sets of data, it is possible to predict the effectiveness of a given countermeasure against a given coherent RGASM.

To enable the design of countermeasures to protect against coherent RGASMs, a variety of tools must first be constructed. Among these is a coherent RGASM test bed to be used for monitoring the behavior of missiles as they are deployed against simulated targets in a laboratory environment. Figure 1.1 is a high level block diagram of the complete RGASM test bed. The coherent RGASM test bed utilizes a preexisting radar consisting of modern coherent RF hardware coupled with a reprogrammable digital back end. This coherent radar is placed within a specially designed anechoic chamber. When the radar is operating, its transmitted energy is received by an array of antennas mounted along one wall of the anechoic chamber. The array of antennas is connected to a complex system which is capable of coherently impressing a target's echo, such as that from a naval vessel with or without a given countermeasure, onto the received energy. The complete signal containing the target echo is then reradiated back through the array so that it can be received and processed by the radar. The radar's behavior is monitored so that its reactions to various targets and/or countermeasures can be accurately quantified.

One component of this test bed is the high speed data capture device (HSCD) for capturing and recording real-time data as it moves through a RGASM's digital processing. This device is shown in Figure 1.1 as the physical link between the radar and the monitoring equipment. The HSCD is coupled directly to the radar's receiver and antenna controller in such a way that the digital signals generated by these components can be monitored without affecting the radar's normal processes. The samples collected by the HSCD are transferred via a high speed data link to an array of computer hard drives for storage and later analysis.

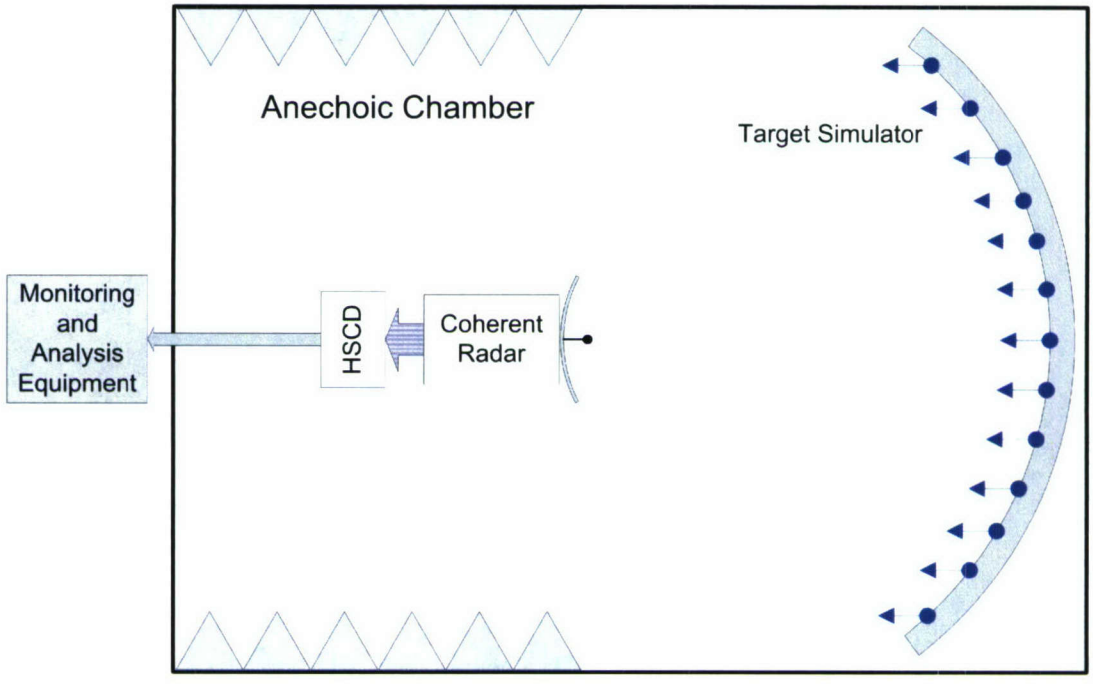


Figure 1.1: Coherent RGASM countermeasure development test bed block diagram.

1.2 Objective and Outline of Thesis

The purpose of this thesis is to document the design and testing of the HSCD. To provide the necessary background, Chapter 2 presents an overview of RGASMs and contrasts coherent and non-coherent approaches. Chapter 3 details the test bed's preexisting radar hardware. Chapter 4 then presents the design of the HSCD hardware, while Chapter 5 details the HSCD's software. Chapter 6 concludes this thesis by discussing some improvements which could be made to the HSCD in future efforts.

1.3 Contribution

The author is responsible for the complete design and assembly of the HSCD. This includes all prototyping and assembly required to create a functional data capture device, as well as writing and testing the embedded software needed for its proper operation.

CHAPTER 2

COHERENT RADAR-GUIDED ANTI-SHIP MISSILE GENERAL SYSTEM OVERVIEW

This chapter describes the basics of a RGASM. First, a basic block diagram of the RGASM is discussed, then the primary differences between coherent and non-coherent processing are compared. A basic digital receiver is then described to illustrate practical implementation of coherent processing. The final section of this chapter briefly examines the advantages coherent pulse doppler radars possess over non-coherent systems in target tracking capability and resistance to jamming.

2.1 RGASM Overview

A block diagram of a basic coherent RGASM is given in Figure 2.1. The specific RGASMs being studied here typically have a coherent pulse doppler radar coupled with control circuitry. The radar measures target echoes in range, azimuth, and elevation and sends this data to the control circuitry which ultimately points the missile in the direction of the target by properly utilizing the control surfaces. The control circuitry implements complex target tracking algorithms which are designed to resist any countermeasures which may cause the missile to miss its intended target. The tracking algorithms implemented in coherent RGASMs have a distinct advantage

over their non-coherent counterparts in that the coherent radar provides additional information about the target that is not available when using a non-coherent radar. To understand these advantages and to illustrate the type of coherent radar used in the RGASM test bed, the rest of this chapter is dedicated to examining coherent pulsed doppler hardware.

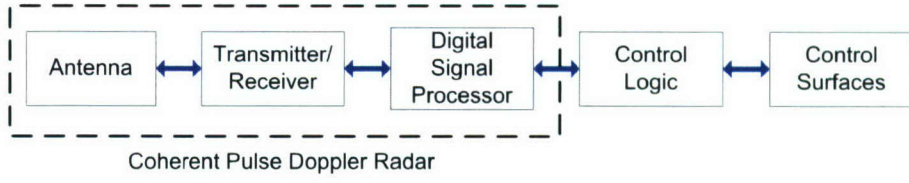


Figure 2.1: Coherent RGASM block diagram

2.2 Coherent Pulse Doppler Hardware

Pulse doppler radars can be distinguished from traditional radars by the three defining characteristics given in Table 2.1 [1]: The first characteristic will be discussed

1. *They utilize coherent transmission and reception; that is, each transmitted pulse and the receiver local oscillator are synchronized to a free-running, highly stable oscillator.*
2. *They use a sufficiently high pulse repetition frequency (PRF) to be ambiguous in range.*
3. *They employ coherent processing to reject main-beam clutter, enhance target detection, and aid in target discrimination and classification.*

Table 2.1: Distinguishing characteristics of coherent radars.

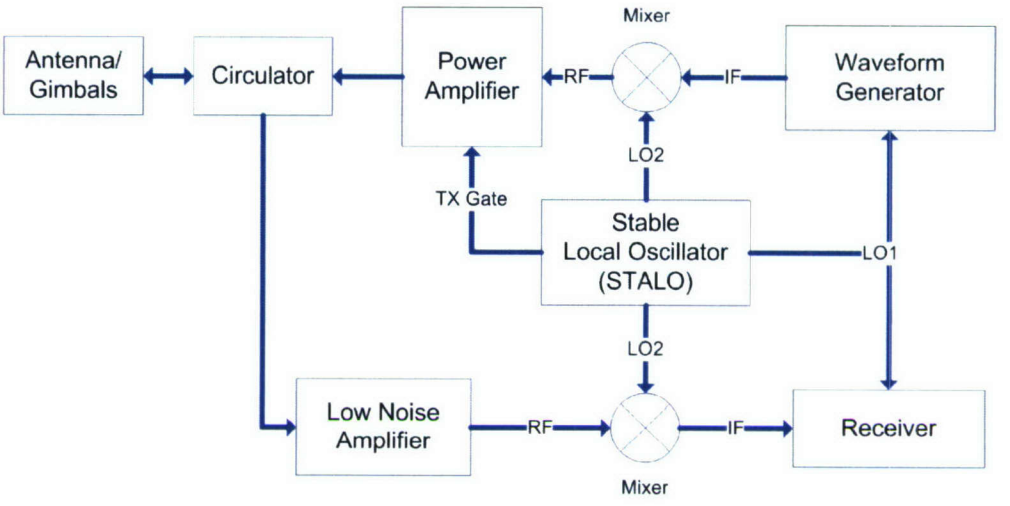
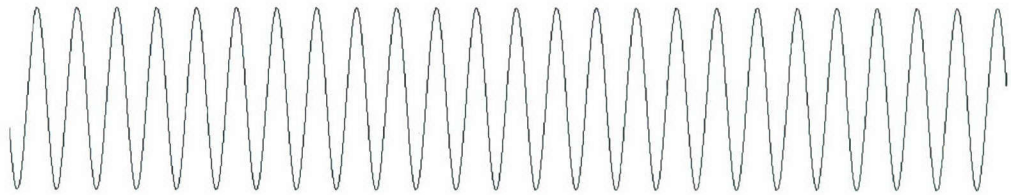


Figure 2.2: Coherent radar block diagram

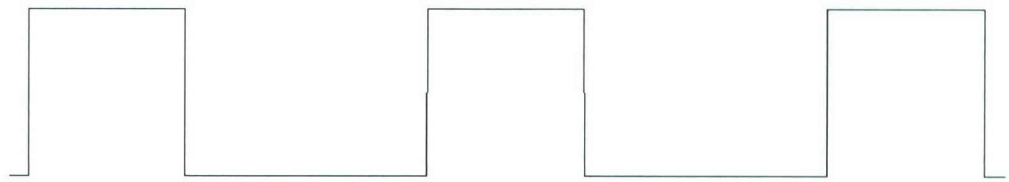
in this section, with the second two characteristics being discussed in subsequent sections.

A high-level block diagram of a basic pulse doppler radar system is shown in Figure 2.2. The layout of the pulse doppler radar is similar in form to traditional non-coherent radars with the exception of a shared stable local oscillator (STALO). The STALO is used to satisfy the first condition of Table 2.1 by ensuring that the transmit and receive chains remain phase-locked. This is accomplished by synchronizing all radar operation with the STALO, including but not limited to the local oscillators (LO) and the transmit gate. An example of a coherent transmitter output is given in Figure 2.3, where Figure 2.3(a) shows a free-running radio frequency (RF) source, and Figure 2.3(b) shows the transmit gate. Both the RF source and the transmit gate are synchronized to the STALO. The transmitted signal is the product of these two signals and is shown in Figure 2.3(c). It is important to note that the RF

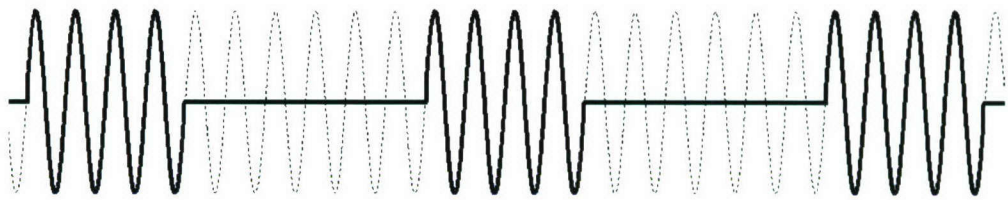
pulses generated by this process are identical and are spaced uniformly by an integer number of wavelengths, the significance of which will be discussed in Section 2.3. In



(a) RF frequency source synchronized to the STALO.



(b) Gating signal corresponding to the pulse repetition frequency.



(c) Coherent pulse train (solid) as generated from the original RF source (broken) multiplied by the gating signal.

Figure 2.3: Creation of a coherent transmit pulse train using an RF source and gating signal which are both synchronized to a stable local oscillator.

contrast, the output of an unsynchronized, or non-coherent, transmit path is given in Figure 2.4. It should be noted that the phasing is inconsistent from pulse to pulse.

The Fourier Transform of a non-coherent pulse train is given in Figure 2.5(a) [2]. The null-to-null width of the main peak in the frequency domain, as marked in the figure, is inversely proportional to length in time of each pulse in the pulse train. In

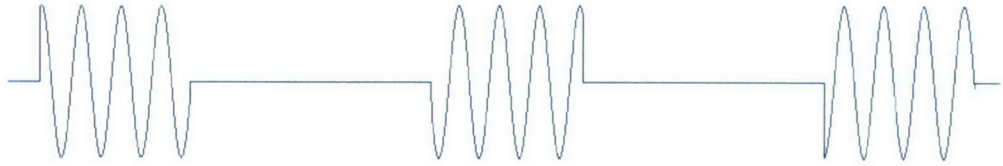


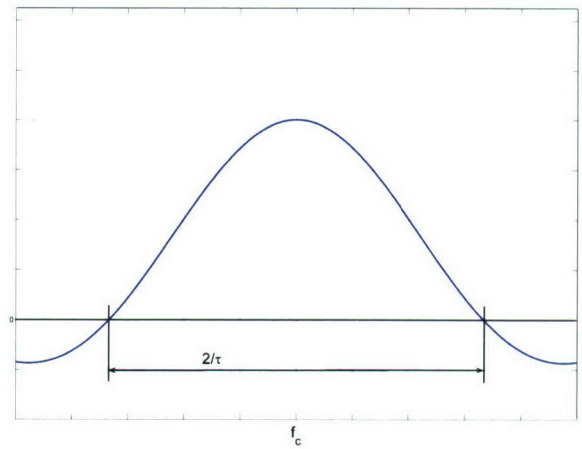
Figure 2.4: Non-coherent transmit pulse train with each pulse starting with a random phase.

contrast, the Fourier Transform of a coherent pulse train is given in Figure 2.5(b) and described by Eq. (2.1) for positive frequencies [2]. In the equation, N is the number of pulses in the train, τ is the width of each pulse, $\omega_c/2\pi$ is the RF carrier frequency, $\omega_o/2\pi$ is the pulse repetition frequency f_{PRF}, and T is the interpulse period.

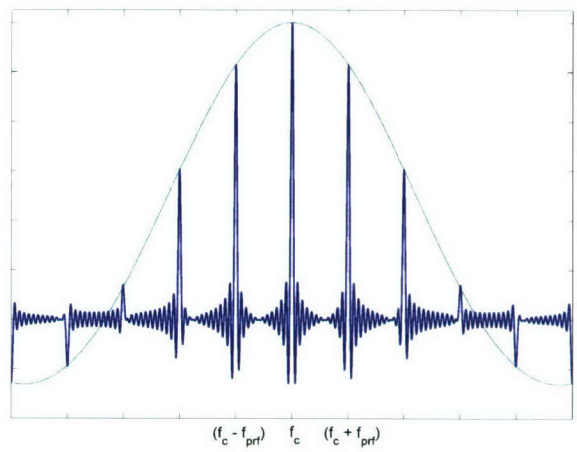
$$F(j\omega) = \frac{A\tau N}{2}\left\{ \begin{array}{l} \operatorname{sinc}\left[(\omega-\omega_c)\frac{NT}{2}\right] \\ +\sum_{n=1}^{\infty} \operatorname{sinc}\left(n\omega_o\frac{\tau}{2}\right)\left[\begin{array}{l} \operatorname{sinc}\left((\omega-\omega_c+n\omega_o)\frac{NT}{2}\right) \\ +\operatorname{sinc}\left((\omega-\omega_c-n\omega_o)\frac{NT}{2}\right) \end{array} \right] \end{array} \right\} \tag{2.1}$$

Comparing the two plots of Figure 2.5, one observes that the non-coherent pulse train distributes energy continuously across the frequency band, where as the coherent pulse train concentrates energy at discrete frequencies. An important effect of this energy concentration is that the magnitude of the envelope of the discrete frequency peaks for the coherent case is greater than the magnitude of the frequency response for the non-coherent case. This characteristic can be exploited by a coherent radar to allow targets to be detected at a greater range than is possible with a non-coherent radar with an equivalent transmit power.

Concentrating the energy of the pulse train into distinct frequency components has several advantages. A receiver capable of exploiting these advantages is required, and such a receiver will be discussed in Section 2.3.



(a) Fourier Transform of a non-coherent pulse train.



(b) Fourier Transform of a coherent pulse train (solid) with the envelope (dotted).

Figure 2.5: Fourier Transform of a non-coherent pulse train versus the Fourier Transform of a coherent pulse train.

2.3 Digital Receivers

A receiver which takes advantage of the coherent pulse doppler hardware described in the previous section will now be examined. A basic diagram of the receiver is given in Figure 2.6. This receiver utilizes a quadrature mixing stage after the standard intermediate frequency (IF) stages to generate in-phase and quadrature channels which are used to provide both the magnitude and phase of the radar returns. These two channels are then sampled by independent analog-to-digital converters (ADC) which are connected to numerical integrators. These numerical integrators simply add a predetermined number of consecutive samples together. The numerical integrator is a type of coherent matched filter which is only made possible through the use of a coherent pulse train. The next stage in the receiver applies pulse compression to the received signal but won't be discussed here for brevity. The pulse compression stage is only included in this basic drawing to parallel the actual radar hardware to be discussed later.

Following the pulse compression stage is the Fast Fourier Transform (FFT) stage which computes a special fast version of the Discrete Fourier Transform of the incoming radar signal. The FFT is used to analyze incoming signals according to their frequency content. The radar return from a moving object will exhibit a small shift in frequency away from the carrier frequency. This is known as the doppler shift. The magnitude of this shift (f_d) in Hertz is a function of the transmitted frequency f_t and the radial velocity v_r of the target according to Eq. (2.2) [3].

$$f_d = \frac{2 f_t v_r}{c} \tag{2.2}$$

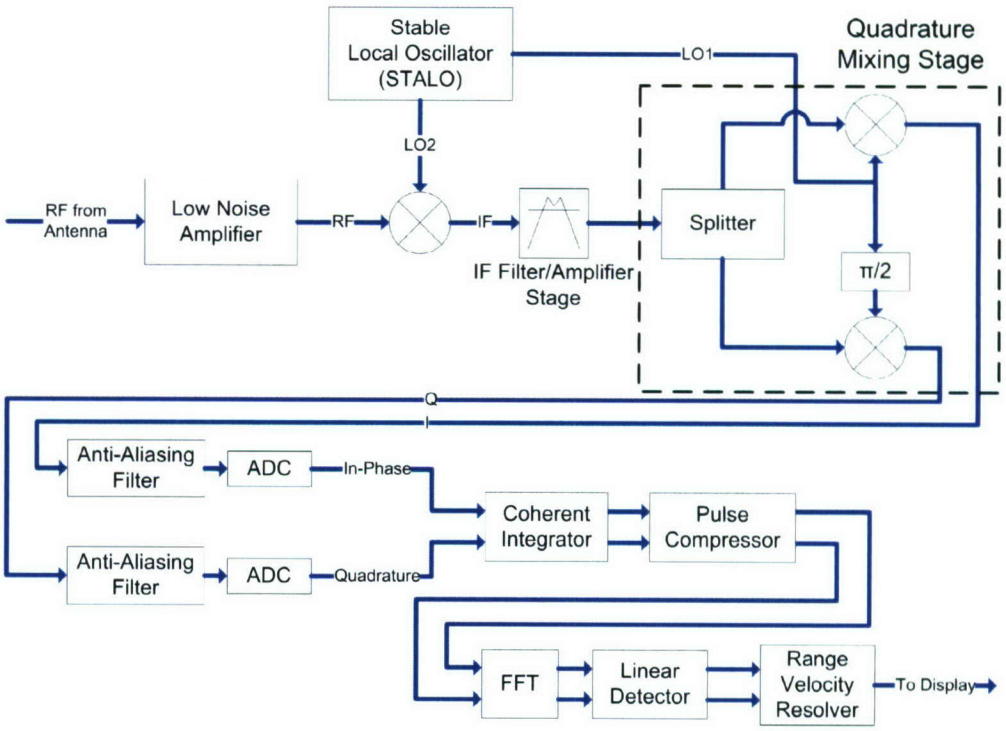


Figure 2.6: Coherent receiver block diagram

The doppler shift is a positive shift in frequency for targets moving toward the radar and a negative shift for targets moving away from the radar.

Recall the second characteristic of pulse doppler radar given in Table 2.1 which states that pulse doppler radars use a sufficiently high pulse repetition frequency (PRF) to be ambiguous in range. This implies that the radar may transmit a pulse before the return from the previous pulse has been measured. In this situation, the range data for the radar returns is ambiguous since the radar is unable to tell from which pulse the return is from. A higher PRF means there is less time between pulses and therefore more ambiguity in the range measurement. However, recall from Section 2.2 that the spectral lines in the Fourier Transform of a coherent pulse train

are separated by the PRF, f_{PRF}. In the frequency domain, a higher PRF means that the spectral lines of the pulse train are spaced further apart. Since the doppler shift of a radar return is a change in frequency, the doppler shift can be said to be unambiguous if it is not great enough to shift past the next adjacent spectral line. Therefore to satisfy this condition, the unambiguous doppler shift must be less than the PRF.

Since the velocity of the radar is usually known, the doppler frequencies of ground clutter in the main lobe of the antenna are known and can be filtered out accordingly. In addition, a sufficiently high PRF guarantees that all encountered doppler frequencies are unambiguous, and therefore a given target's radial velocity can be directly calculated. The ability to calculate a target's velocity in addition to its range allows a pulse doppler radar to more accurately classify a target than a non-coherent radar. This, coupled with ground clutter mitigation, corresponds to the third characteristic of pulse doppler radars in Table 2.1.

Following the FFT is a linear detector which decides whether or not targets exist in the radar return. Once a target has been detected, it is passed to the range ambiguity resolver stage. As mentioned earlier, unambiguous velocity usually implies ambiguous range measurements. This stage is required to determine target range information in a radar system where the target's velocity (doppler shift) is unambiguous. The target's true range is decoded using its range information from several PRF's as described by Stimson [2].

2.4 Coherent Versus Non-Coherent Target Tracking

Coherent radar systems have distinct advantages over their non-coherent counterparts in target tracking applications. Typically, coherent processing yields greater range and range resolution while employing a lower transmit power [4]. A lower peak transmit power makes a coherent RGASM less likely to be detected by its target. In the event that a target is alerted to the RGASM's presence, increased range resolution combined with target velocity sensing capabilities decrease the likelihood of the coherent RGASM being defeated by traditional countermeasures. Therefore, countermeasures geared toward defeating coherent RGASMs need to be designed and tested, and the high speed data capture device described in this thesis will play a small part in that larger task.

CHAPTER 3

TARGET RADAR HARDWARE ARCHITECTURE OVERVIEW

The radar system used in the coherent seeker test bed is a coherent pulsed doppler radar with a receiver similar to that shown in Figure 2.6. It consists of two main components: a processing unit, which contains all of the RF and signal processing hardware, and a mechanically scanned static phased array antenna. The processing unit will be examined closely here with special attention being given to the receiver and the antenna processor interface (API) since these components generate all of the data to be captured by the HSCD.

3.1 Receiver Overview

The radar is a coherent doppler monopulse system consisting of sum and difference channels. The theory of monopulse operation will not be discussed here except to point out that it requires two independent receivers. The start of the receive chain consists of two RF receivers, one for each channel, which handle the coherent downconversion from RF to baseband before sampling the data. The in-phase and quadrature signals at the output of the quadrature mixing stage can be sampled using either 11-bit analog-to-digital converters (ADC) running at 1MHz or 8-bit ADCs

running at 20MHz. For the purpose of this thesis, it is assumed that the receivers are always using the 8-bit ADCs running at 20MHz. Following the ADCs are arithmetic logic units (ALU) which can integrate up to 16 consecutive samples. The outputs of the ALUs are always 16 bits regardless of the ADC being used. These 16-bit values are buffered in memory before being transferred to the pulse compression unit (PCU). After pulse compression operations have been performed, the two channels (sum and difference) of in-phase and quadrature data are forwarded on to the rest of the digital signal processing chain.

3.2 Antenna Processor Interface (API) Overview

The antenna processor interface is a special subsystem within the radar which controls the antenna's gimbals according to commands given by the radar's central processor. Within the gimbals are resolvers and tachometers which constantly measure the position and velocity of the antenna in azimuth and elevation. These measurements are compared in the API to the azimuth and elevation position, velocity, and acceleration requested by the central processor. The differences between the requested and actual positions, velocities, and accelerations are fed into a complex control loop to properly operate the servo motors within the gimbals which ultimately point the antenna in the desired direction.

To maintain smooth antenna motion, the central processor updates the API with new antenna position requests at regular intervals. The API, in turn, updates the central processor with the actual measured antenna position so that this information can be used in other parts of the digital signal processing chain. This two-way communication occurs over a data bus which is shared by the API with other subsystems

within the radar. Each subsystem connected to the data bus is associated with its own specific set of memory addresses to facilitate easy data transfer between each subsystem and the central processor. This data bus will be examined more closely in Section 3.3.3.

3.3 High-Speed Data Capture Locations

As previously mentioned, the sampled in-phase and quadrature data from the receivers as well as the antenna position data from the API must be captured in real-time. The strategy for accomplishing this task is based on the organization of the data flow within the radar itself. The data flow, in turn, is based on how the radar scans for targets. Therefore, it is first important to establish some basic terminology to describe the radar's scanning behavior before examining data transfers within the radar in greater detail.

3.3.1 Radar Scanning Terminology

The most basic behavior of the radar is a scan of the antenna from left to right or from right to left. This motion is called a bar, and the bar is broken up into smaller pieces of consistent lengths of time called communication intervals (CI). Each CI within the bar has a characteristic PRF which is usually varied from CI to CI to allow for the resolution of range ambiguity as mentioned in Section 2.3. Since every CI has a characteristic PRF, each pulse within the CI is separated from the next by a constant length of time called the inter-pulse period or IPP. Each IPP is broken up into a number of range samples (RS) which correspond to a single output of the receivers' coherent integrators. The number of range samples within an IPP depends on three factors: the PRF which determines the length of the IPP, the sampling rate

of the ADCs in the receiver, and the number of coherently integrated samples after the ADCs. Each range sample represents a point in time, or a segment of time if several consecutive samples are coherently integrated, where a target's radar return may be present in the data. The organization of the CI is very important in terms of understanding how the range samples are collected and processed, and this will become apparent in the following section.

3.3.2 Receiver In-Phase and Quadrature Data Transfers

The radar's two receivers change parameters on a CI-to-CI basis according to the PRF, desired number of coherent integrations, and other factors. To facilitate this, the receivers are reset at the end of every CI to prepare for operation with new parameters during the subsequent CI. During the first IPP of a new CI, the receivers generate range samples corresponding to the first transmitted pulse. The range samples are buffered temporarily in two first-in-first-out (FIFO) memories per receiver, separating the in-phase and quadrature samples between them. During the next IPP, the in-phase and quadrature data from the first receiver are transferred to the pulse compression unit, followed by the in-phase and quadrature data from the second receiver. While these data transfers are being performed, the receivers are simultaneously generating range samples for the current IPP. This process is shown in Figure 3.1. In this figure and in future figures, the sum and difference receivers will be addressed as Channels A and B, respectively. It is important to note that all in-phase range samples are transferred to the PCU for one channel followed by all quadrature range samples for the same channel. This process is then repeated for the second channel. It is interesting that while the receivers generate range samples

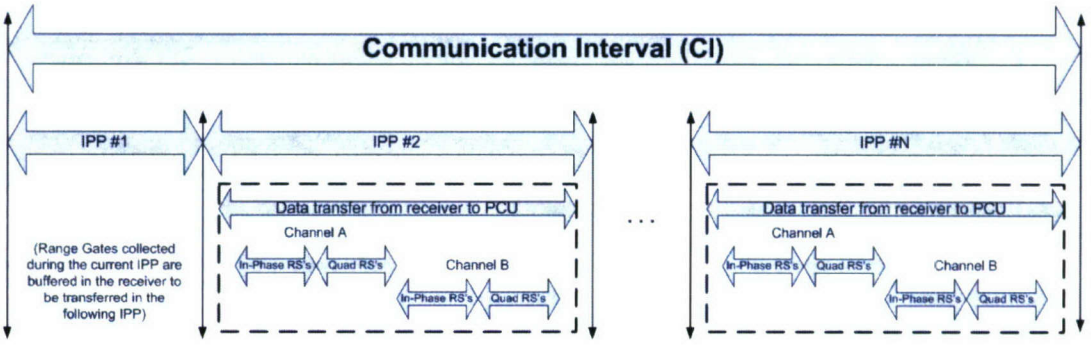
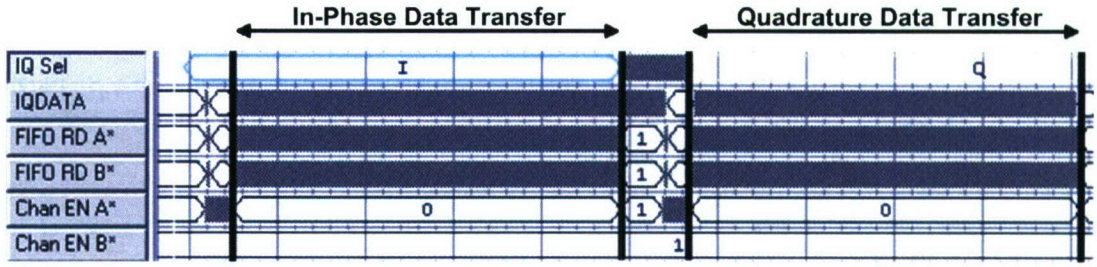


Figure 3.1: Communication interval broken down by inter-pulse period showing the transfer of in-phase and quadrature data from the receiver FIFOs to the PCU.

during the last IPP of a CI, they are never actually transferred out of the receiver and are effectively lost at the start of the next CI.

The receivers and PCU are constructed on physically separate circuit boards which are connected to a common backplane. Therefore, it is possible to monitor the in-phase and quadrature range samples as they are transferred from the receivers to the PCU without actually disrupting the processes occurring within the radar. A Tektronix TLA715 logic analyzer was used to examine the control signals present on the backplane in order to develop an understanding of how the data is transferred. Connecting the logic analyzer to the backplane without removing any of the hosted circuit boards is possible through the use of a special backplane which features connectors on one side and special pins on the reverse side. This backplane was specifically designed for use during the radar's research and development phase by its manufacturer so that logic analyzers could be utilized to debug the radar's operating software.

A screen shot of the logic analyzer data is shown in Figure 3.2 which illustrates the range samples transferred from the receiver to the PCU during a single IPP.



Label:	Fuction:
IQ_Sel	Signifies whether data present on the bus is In-Phase or Quadrature
IQDATA	Actual 16-bit value of the data
FIFO_RD_ A*	Read strobe for Receiver A
FIFO_RD_B*	Read strobe for Receiver B
Chan_EN_A*	Receiver A Enable
Chan_EN_B*	Receiver B Enable

Figure 3.2: Logic analyzer capture showing the transfer of in-phase and quadrature data from the receiver to the pulse compression unit over the backplane with labels identified.

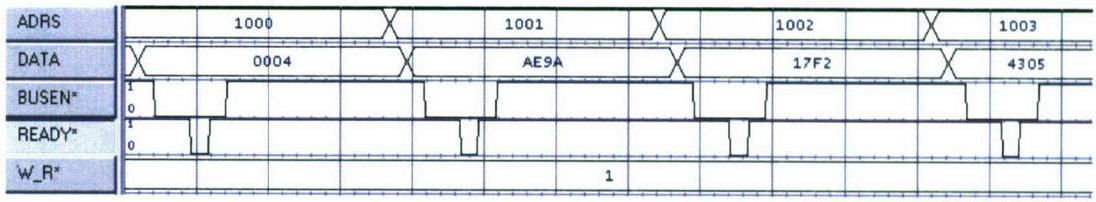
The table below the screen shot lists the function of each signal under observation. The black bars were added later to clearly distinguish the beginning and ends of each transfer. The gray areas in the screen shot indicate periods where the signals under observation are transitioning too rapidly to be drawn on the screen. A close examination of the IQ_Sel signal between the black marks enforces the fact that all in-phase range samples for an IPP are transferred to the PCU followed by all quadrature range samples. These samples are transfered on the active edge of the FIFO_RD_A or B strobes, which serve to clock the data out of the receivers' FIFOs and into the PCU. Figure 3.2 was produced during a radar mode which does not utilize monopulse processing. Therefore, only data from the sum receiver is transferred to the PCU, which can be seen since only the Ch_EN_A* signal is active and not Ch_EN_B*. For

modes which require monopulse capabilities, the process shown in the figure would be repeated twice per IPP, once for Channel A and once for Channel B.

3.3.3 API Data Transfers

The API typically communicates with the radar's central processor once per CI over the shared bus. As mentioned earlier, the scope of this communication includes both desired and actual antenna position, velocity, and acceleration. In addition, the API also provides the central processor with the errors between the measured and desired position and velocity, as well as the results from built-in self-test circuitry which constantly monitors the health of the API and antenna.

The API and the central processor circuit cards are connected to the same backplane previously mentioned which allows for passive monitoring of the shared data bus by a logic analyzer. A screen shot of the logic analyzer data is shown in Figure 3.3 which illustrates the basic operation of the data bus. A list of the observed signals and their functions is given below the screen shot. The central processor acts as the bus master by controlling the Address (ADRS in the figure) lines and the Bus Enable (BUSEN*) lines. Each subsystem on the bus is assigned a master address which is contained in the most significant nibble of the 16-bit ADRS. Therefore, to initiate a data transfer on the bus, the central processor asserts an address to be read or written and activates the BUSEN* signal. The subsystems attached to the bus then observe the most significant nibble on the address line to decide if they are involved in the data transfer. The API is assigned to master address 1. Therefore, any data transfer on the bus with an address of $10XX requires the API's attention. This address is represented here in hexadecimal where 'X' signifies an unknown digit. When the



Label:	Fuction:
ADRS	16-bit address bus value
DATA	Actual 16-bit value of the data on the bus
BUSEN*	Bus enable
READY*	Bus ready
W_R*	Write (high) or read (low) cycle select

Figure 3.3: Logic analyzer capture showing the transfer of antenna data from the API to the radar's central processor over the backplane with labels identified.

BUSEN* line is pulled low by the central processor with an address of $10XX, the API places the proper data onto the bus or reads the data from the bus, according to the W_R* signal, and activates the READY* signal. The READY* signal notifies the central processor the data has been handled appropriately, and that the bus is free again for another transfer.

3.4 Conclusion

The in-phase and quadrature range samples, in addition to all of the data transfered to and from the API, must be captured in real time without interrupting the radar's processes. Since the design of the radar utilizes a common backplane architecture, the required data can be captured successfully by special hardware which is designed to physically connect to the radar's special development backplane. This

hardware must be smart enough to passively decode the bus communications, reformat the data, and forward it to a data recorder. The HSCD hardware is designed to perform these tasks and will be discussed in Chapter 4.

CHAPTER 4

HIGH SPEED DATA CAPTURE HARDWARE IMPLEMENTATION

Chapter 3 discussed some aspects of the radar architecture to which the HSCD will interface, specifically the receivers (and PCU) and the API. The HSCD is designed to physically connect to the backplane of the radar in order to monitor these components, much like a digital logic analyzer, without modifying the existing radar hardware in any way or causing any degradation in signal quality on the backplane. How the HSCD fits into the overall data recording system can be seen in Figure 4.1. The HSCD intercepts bus transfers to and from API and PCU, bundles the data it accumulates with other required information, and then sends it to a VME-based parallel-to-fiber optic converter. This parallel-to-fiber converter is capable of transmitting the captured data serially at rates up to 2.5 gigabits-per-second (Gbps) over a long (up to 300m) fiber optic cable. A data recorder, which is essentially a computer containing a large array of harddrives and a PCI-based fiber optic receiver connected to the parallel-to-fiber optic converter, produces a real-time display of the data while simultaneously writing it to the disks.

The high speed data capture device hardware design can be broken down into three major sections: the connection to the radar's backplane, use of a Field Programmable

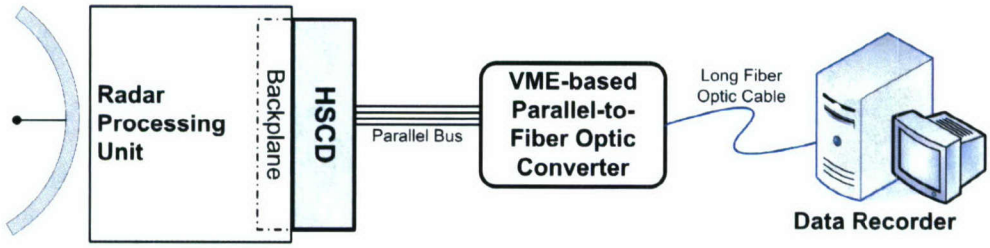


Figure 4.1: Complete high speed data recording system.

Gate Array (FPGA), and the connection to the parallel-to-fiber optic converter. Each of these sections will be discussed in detail followed by a close examination of the how these sections are integrated onto the HSCD printed circuit board (PCB).

4.1 Radar Backplane Description

A diagram of the backplane to which the HSCD connects is shown in Figure 4.2. This figure shows the relative locations of the API and PCU card slots with respect to one another as viewed from the exposed rear portion of the backplane. The radar's processing cards, including the API and PCU, are attached to the other side of the backplane and are concealed within the radar's chassis. Access to the rear of the backplane can be gained by simply removing a metal cover from the chassis.

The data probe points on the rear of the special development backplane consist of three rows of 100-mil spaced pins, with 56 pins in each row for a total of 168 pins per card. All the signals of interest routed across the backplane are transistor-transistor logic (TTL) 5V signals. Several important signals have already been mentioned in Chapter 3 while discussing the operation of the receivers and the API. The signals which are accessible through the PCU connector are listed in Table 4.1, while those

accessible through the API connector are listed in Table 4.2. In addition to these signals, 5V power and ground pins are present to supply the API and PCU cards. These pins also will provide a convenient source of power for the HSCD while the radar is operating.

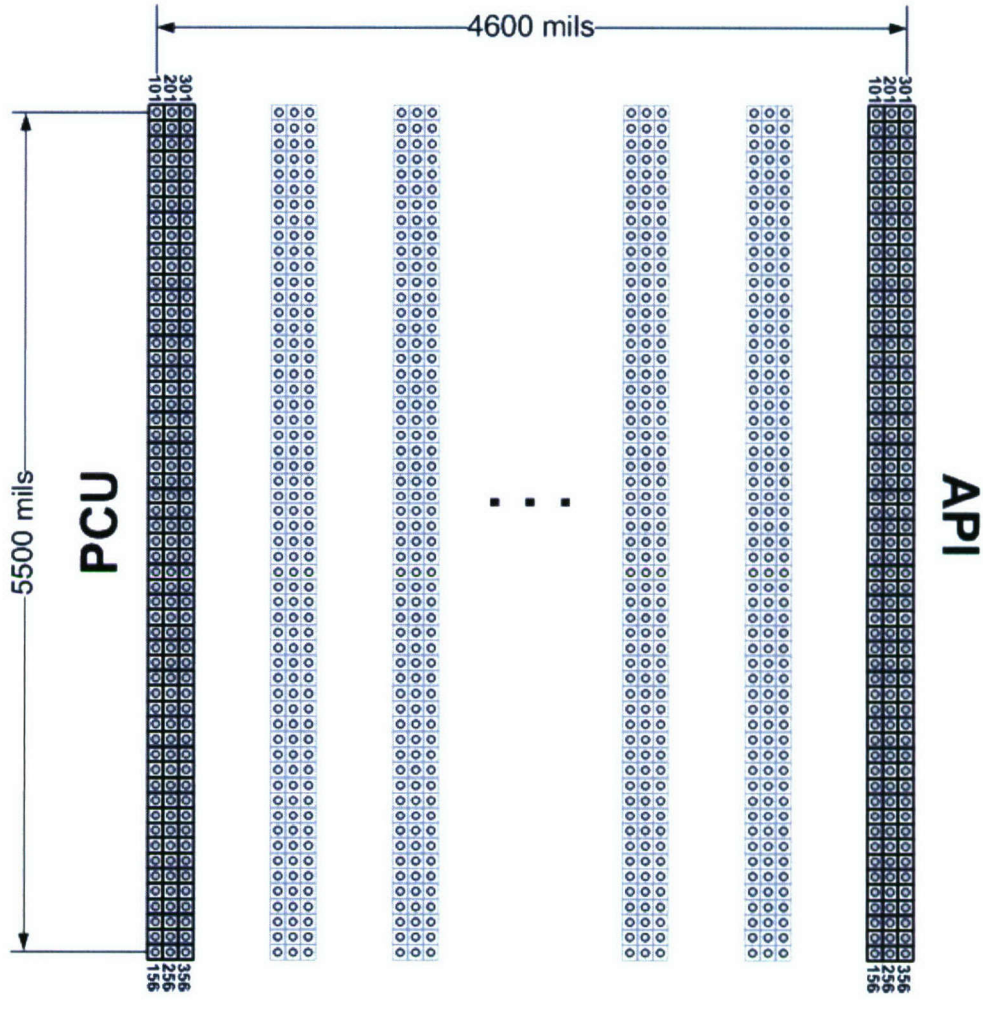


Figure 4.2: Radar backplane showing location and dimensions of the PCU and API card connectors as viewed from the rear of the radar module. Additional connectors which exist to the left of the PCU and to the right of the API are not shown.

PCU:			
Pin Name:	Slot:	Pin:	Notes:
IQDATA00	PCU	150	IQ Data from receiver (LSB)
IQDATA01	PCU	250	IQ Data from receiver
IQDATA02	PCU	350	IQ Data from receiver
IQDATA03	PCU	151	IQ Data from receiver
IQDATA04	PCU	351	IQ Data from receiver
IQDATA05	PCU	152	IQ Data from receiver
IQDATA06	PCU	252	IQ Data from receiver
IQDATA07	PCU	352	IQ Data from receiver
IQDATA08	PCU	153	IQ Data from receiver
IQDATA09	PCU	253	IQ Data from receiver
IQDATA10	PCU	353	IQ Data from receiver
IQDATA11	PCU	154	IQ Data from receiver
IQDATA12	PCU	354	IQ Data from receiver
IQDATA13	PCU	155	IQ Data from receiver
IQDATA14	PCU	255	IQ Data from receiver
IQDATA15	PCU	355	IQ Data from receiver (MSB)
FIFORdSUM+	PCU	346	SUM Ch. FIFO read strobe (+)
FIFORdSUM-	PCU	347	SUM Ch. FIFO read strobe (-)
FIFORdDIFF+	PCU	146	DIFF Ch. FIFO read strobe (+)
FIFORdDIFF-	PCU	147	DIFF Ch. FIFO read strobe (-)
IQHeader	PCU	348	Start of Data from Receiver
IQSel	PCU	247	0 = In Phase, 1 = Quadrature
Chan_EnSUM*	PCU	246	Active LOW
Chan_EnDIFF*	PCU	249	Active LOW
RCVDELTC*	PCU	127	Start of IPP
RCVLENTC*	PCU	128	End of IPP
FIRSTIPP*	PCU	129	Indicates first IPP of a CI
LASTIPP*	PCU	130	Indicates last IPP of a CI
IQSat	PCU	148	Indicates pre or post integration saturation
20MCLK	PCU	225	20 MHz Clock

Table 4.1: Backplane pin assignments used for PCU data capture.

API:			
Pin Name:	Slot:	Pin:	Notes:
BUSEN*	API	133	Bus Enable
READY*	API	333	Bus Ready
W_R*	API	334	Write (high) or read (low) cycle select
ADRS00	API	136	Bus address (LSB)
ADRS01	API	336	Bus address
ADRS02	API	137	Bus address
ADRS03	API	237	Bus address
ADRS04	API	337	Bus address
ADRS05	API	238	Bus address
ADRS06	API	138	Bus address
ADRS07	API	338	Bus address
ADRS08	API	140	Bus address
ADRS09	API	240	Bus address
ADRS10	API	340	Bus address
ADRS11	API	141	Bus address
ADRS12	API	241	Bus address
ADRS13	API	341	Bus address
ADRS14	API	142	Bus address
ADRS15	API	342	Bus address (MSB)
DATA00	API	143	Bus data (LSB)
DATA01	API	243	Bus data
DATA02	API	343	Bus data
DATA03	API	144	Bus data
DATA04	API	244	Bus data
DATA05	API	344	Bus data
DATA06	API	145	Bus data
DATA07	API	345	Bus data
DATA08	API	146	Bus data
DATA09	API	246	Bus data
DATA10	API	346	Bus data
DATA11	API	147	Bus data
DATA12	API	247	Bus data
DATA13	API	347	Bus data
DATA14	API	148	Bus data
DATA15	API	348	Bus data (MSB)

Table 4.2: Backplane pin assignments used for API data capture.

The majority of signals routed across the backplane are single-ended, meaning they are referenced to a common ground and are not differential signals. Careful examination of the radar's processing cards indicate that every single-ended signal of interest is generated and terminated by standard TTL components. However, the FIFO read strobes for both the sum and difference channel receivers are differential signals which require both a positive and negative connection. A differential line receiver is used to translate these differential pairs into standard single-ended signals which can then be used with standard TTL devices.

4.1.1 Data Transfer Speed Requirements

The FIFO read strobes generated by the PCU to clock data out of the receivers operate in bursts with a clock rate of approximately 3.8 MHz. During these bursts, sixteen bits of data are transferred from the receiver to the PCU on every read strobe which translates to approximately 60 megabits per second (Mbps), or 7.6 megabytes per second (MBps). Figure 4.3 illustrates the nature of these data transfers on an IPP-to-IPP basis. The in-phase and quadrature data transfers in the figure occur at the beginning of every IPP. The number of data transfers which occur is determined by the number of range samples collected during the previous IPP as discussed in Section 3.1, and as such is constant throughout the CI. From the figure, it is evident that the data bus is dormant for a large portion of the IPP. This free time is used by the radar to apply pulse compression to the range samples and then to complete all other signal processing tasks before the beginning of the next IPP.

As mentioned in Section 3.3.3, the API typically communicates with the central processor approximately once per CI, with additional communication occurring as

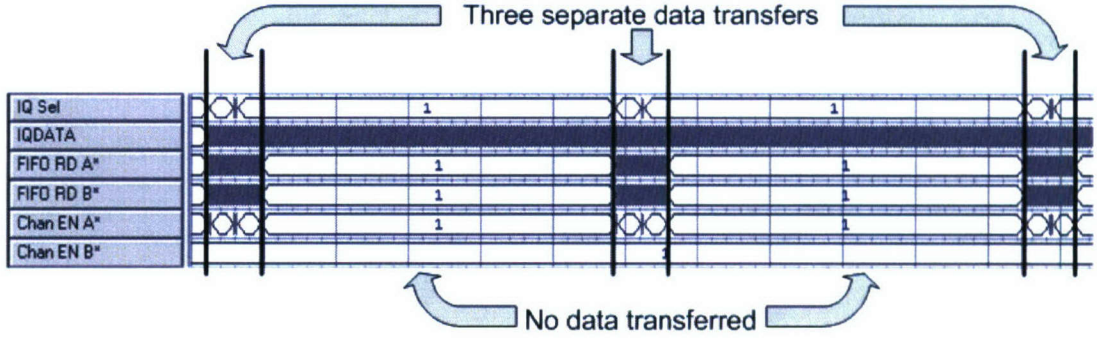


Figure 4.3: Logic analyzer screen shot of in-phase and quadrature data transfers over the course of three separate IPPs.

required to complete some signal processing tasks. Standard communication entails the API either receiving requested antenna position parameters or sending the current antenna position parameters. The data transfers contain at most a total of sixteen 16-bit words in the worst case scenario. The API either acknowledges or initiates these data transfers with a clock rate of 1.11 MHz which equates to approximately 17.76 megabits per second in each burst.

The in-phase and quadrature range samples transferred from the receivers to the PCU comprise the bulk of the data to be intercepted by the HSCD. Transfer rates discussed previously occur only for short periods of time at the beginning of each IPP, with true data rates being slightly lower over longer periods of time. Actual data rates over a larger timescale such as a CI will be directly proportional to the PRF, since this determines the length of the IPP and the number of range samples transferred during the IPP. In addition, modes which do not utilize monopulse processing will only transfer half the total number of range samples to the PCU in a given IPP compared to a monopulse mode which would require range samples from both the

sum and difference channel receivers to be transferred. The API data transfer bursts contribute only a small fraction of the total data to be handled by the HSCD due to their brevity and relatively wide spacing in time. However, since the API and PCU operate fully independently of one another, it is possible for both buses to be active simultaneously. In this case, their combined data rate could reach 77.76 megabits per second for short periods of time.

For the HSCD to operate in real time without significant on-board data buffering, it must be capable of processing the data from the API and PCU simultaneously. Furthermore, the interface between the HSCD and the parallel-to-serial converter mentioned earlier must have enough bandwidth to handle a peak data rate of 77.76 megabits per second generated by the radar plus additional data generated by the format used to identify the captured information. This format will be discussed in Section 5.2 while the interface to the parallel-to-serial converter will be discussed in the following section.

4.2 Field Programmable Gate Arrays

Data transfers involving the API and PCU occur independently of one another and often at the same instant of time. For the HSCD to effectively monitor these systems simultaneously, it must be capable of performing parallel tasks involving simple calculations and data buffering. A field programmable gate array (FPGA) is the ideal choice for this application.

FPGAs are capable of carrying out simultaneous parallel processes by utilizing built-in random access memory (RAM) blocks and embedded array blocks (EAB) to implement complex logic functions. Altera is a major manufacturer of programmable

logic devices including FPGAs and has readily supplied The Ohio State University with the software and programming tools necessary to implement solutions using their devices. Because of this exposure to Altera's products, their FPGAs were an obvious choice for use in the HSCD.

4.2.1 Power Supply Requirements

Several possible choices were examined for the model of FPGA to be used in the HSCD. The Altera Flex10K® FPGA, while not the most modern offering, was chosen for several reasons. The first and foremost reason is that it is based on a 5V core with the possibility of either 3.3V or 5V input/output (I/O) levels. The radar backplane supplies a convenient 5V regulated power source which can be used to directly power the Flex10K®, greatly simplifying the overall board design. Additionally, a requirement of the HSCD is that it will not interfere with the radar in any way. The Altera Flex10K® supports hot-socketing which allows signals to be driven into the FPGA before it is powered up without damage. More importantly, the Flex10K® does not drive out during power up [5]. These features are important to limit the HSCD's impact on the radar's backplane during times of power instability such as when the radar is powering up and powering down.

4.2.2 Input-Output Voltage Standards

Other factors were considered when choosing an FPGA such as I/O standards, number of I/O pins available, and the physical device package. Since the radar backplane implements primarily 5V TTL interfaces and the Altera Flex10K® supports 5V I/O as mentioned earlier, the radar's signals can be fed directly into the FPGA

without the need for buffers or level converters. This simplifies the board layout and eases the load on the host radar's power supply.

4.2.3 Device Packaging

The FPGA must have enough available I/O pins to access all of the signals listed in Tables 4.1 and 4.2. In addition, 36 pins must be available to drive the parallel-to-fiber optic converter interface discussed in Section 4.3. The Altera Flex10K® is available in a 240-pin Power Quad Flat Pack (RQFP) package which advertises 189 total user I/O pins [6]. In reality, the number of usable I/O pins is slightly lower as a portion of the 189 I/O pins are reserved for programming pins, dedicated inputs, and dedicated clock pins.

The availability of of the Flex10K® in the 240-pin RQFP package as opposed to other packages such as the ball grid array (BGA) is important because this affords the possibility hand soldering the chip onto a printed circuit board. This greatly decreases development time and costs since an outside company is not required to mechanically place and solder the part. Indeed it has been shown that the RQFP package can be reliably placed by hand since the pins are easily accessible, as opposed to BGA packages for which hand soldering is impossible.

4.2.4 In-Circuit Programmability

The Altera Flex10K® FPGA is a RAM-based device. As such, it is inherently faster than EEPROM based complex programmable logic devices (CPLD) offered by Altera such as the MAX® series devices. However, since the Flex10K® is based on volatile memory, it must be reprogrammed every time it is powered on. The HSCD implements two separate and independent methods of programming the Flex10K®.

The method which will be discussed first is the passive serial (PS) method which enables a special serial memory chip to automatically program the FPGA on power up. The second method is the Joint Test Action Group (JTAG) connector which allows the FPGA to be directly programmed by a computer using an Altera programming cable. The JTAG method is also capable of programming the special memory device, and as such will be discussed second.

Passive Serial

The passive serial configuration method allows for an FPGA such as the Flex10K® to be programmed serially by microcontrollers, special serial memory devices, or even other EEPROM-based FPGAs. The HSCD utilizes a special Altera serial configuration device, the EPC2. The EPC2 is a flash memory with special circuitry which automatically serially transfers the configuration stored in its memory to any connected FPGA at power up. The EPC2 is programmed with a special configuration file generated by Altera Quartus® software. This file is programmed into the EPC2 using the JTAG interface.

JTAG

The JTAG interface, as specified by the IEEE 1149.1 Standard, is commonly built into microprocessors and programmable logic devices as a method of in-circuit debugging and programming. The JTAG interface is basically a state machine which allows for serial data to be written to or read from the device under test. The JTAG interface consists primarily of four signals: TCK, TMS, TDI, and TDO. The TCK signal acts as the clock for the serial data and built-in state machine. The TMS signal

determines how the JTAG state machine will behave. The TDI and TDO signals are serial lines which carry data into and out of the device under test.

Altera uses JTAG to implement several types of functionality in their FPGAs ranging from programming to analysis of embedded logic. Access to the JTAG functionality in an FPGA requires the implementation of a JTAG connector on the printed circuit board. Altera manufactures the USB Blaster download cable which connects the JTAG port to a computer's USB port. Table 4.3 [7] describes the USB Blaster's pin assignments. The pins in the table include the four JTAG signals mentioned earlier in addition to power and ground connections. The JTAG port implemented on the PCB connects the power and ground pins from the USB Blaster to the device's power and ground supplies and also connects the four JTAG signals from the USB blaster to the corresponding JTAG pins on the device. No other connections are needed.

Altera USB Blaster JTAG Pin Assignments		
Pin:	JTAG:	Description:
1	TCK	Clock signal
2	Ground	Signal Ground
3	TDO	Data from device
4	V_{CC}	Target power supply
5	TMS	JTAG state machine control
6	-	No connect
7	-	No connect
8	-	No connect
9	TDI	Data to device
10	Ground	Signal Ground

Table 4.3: Altera JTAG implementation for the USB Blaster download cable.

An important feature of the JTAG interface is the ability to chain devices together so that several devices can be accessed from the same JTAG connector. To access more than one device from the same connector, the TCK and TMS signals must be connected to each device in the chain. The TDI signal is connected to the first device in the chain and its TDO signal is connected to the TDI of the second device in the chain and so on. Finally, the TDO from the last device in the chain is fed back to the JTAG port completing the chain.

The Altera Quartus® software provides access to the JTAG port from a computer and has the ability to scan the JTAG chain and generate a list of the devices connected. Each device in the list can then be accessed or programmed accordingly. The HSCD implements a JTAG port for the purpose of programming both the FPGA and the EPC2 configuration device. A diagram of how this is accomplished is shown in Figure 4.4 [8]. In the figure, the EPC2 and FPGA are connected to form a JTAG chain which is accessible by the JTAG port. In addition to the JTAG chain, the PS connections between the EPC2 and the FPGA are also shown. A simple buffer chip, left out of the diagram for simplicity, is required to properly drive the JTAG chain when more than one device is connected to it. In the Altera Quartus® software, the EPC2 is detected as the first device in the chain while the FPGA is detected as the second device, as arranged in Figure 4.4. The RAM-based FPGA can be programmed directly but loses its programming once the power has been removed. The EPC2 can be programmed directly as well. However, its flash memory maintains its programming even when power has been removed. Once the HSCD is powered back on, the EPC2 automatically programs the FPGA over the PS connection with the data contained in its nonvolatile memory. Using this system, the HSCD FPGA can

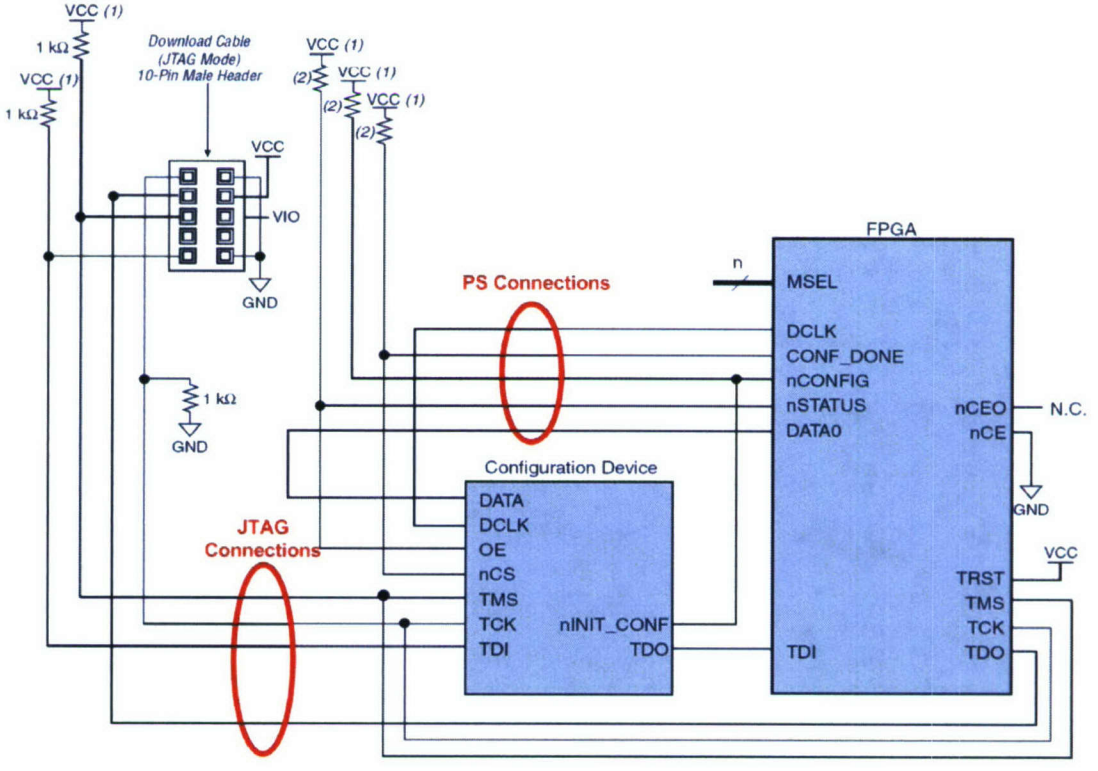


Figure 4.4: Combination of JTAG and Passive Serial configuration scheme implemented within the HSCD (red marking added for clarity).

be easily reprogrammed using the JTAG port to quickly develop the required operational firmware. Once the firmware has been fully developed, it can be stored in the EPC2 to program the FPGA whenever power is applied.

4.3 Parallel-to-Fiber Optic Converter Interface

The HSCD combines the range samples and antenna positions it collects from the radar with other identifying information. The complete data set is transferred to the VME-based parallel-to-fiber optic converter shown in Figure 4.1 over a parallel data bus. This parallel data bus is a 6-foot-long, 60-conductor ribbon cable with a characteristic impedance of 50 ohms. The ribbon cable terminates into 3.3V tristate bus transceivers within the parallel-to-fiber optic converter whose direction is controlled by a Xilinx Virtex FGPA. For this application, the direction of the bus transceivers will always be set to read data from the HSCD.

The ribbon cable connector is of the AMP® System 50 variety with the conductors adhering to the assignments listed in Table 4.4. As can be seen in the table, there are 60 individual conductors contained within the cable. Thirty-six of those are available for data, data-strobe, or clock functionality. The other conductors are permanently tied to ground. The ground conductors are purposefully placed between the signal conductors wherever possible to cut down on crosstalk which could possibly corrupt the data. This configuration will be more apparent in Section 4.4.3 where the physical layout of the connector on the HSCD will be discussed.

Since the Flex10K® will be powered from the radar's 5V supply and will be configured for 5V I/O, special integrated circuits (IC) are used to convert the FPGA's outputs to a level which will not damage the parallel-to-fiber optic converter's inputs.

AMP® System 50 Connector Pin Assignments	
Function:	Connector Pin Number:
Data(35...0)	2, 3, 4, 5, 7, 8, 9, 10, 12, 13, 14, 15, 17, 18, 19, 20, 22, 23, 24, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, 49, 51, 53, 55, 57
Ground	1, 6, 11, 16, 21, 26, 28, 30, 32, 36, 38, 40, 42, 44, 46, 48, 50, 52, 54, 56, 58, 60
No Connection	34, 59

Table 4.4: AMP® System 50 connector pin assignments for the ribbon cable linking the HSCD to the parallel-to-fiber optic converter.

The 74LVTH245A IC 3.3V octal bus transceiver is designed to bidirectionally connect a 5V data bus to a 3.3V data bus. The 74LVTH245A can achieve this utilizing only a single 3.3V supply because its I/O pins are 5V-tolerant [9]. Therefore, the 5V outputs from the FPGA can be applied directly to one side of the bus transceiver with the outputs of the transceiver being used to drive the parallel cable connected to the parallel-to-fiber optic converter.

Due to the length of the ribbon cable, special precautions must be taken to ensure the integrity of the data transfered over the parallel data bus. A series termination technique will be used to minimize reflections due to impedance mismatches between the source, transmission line, and load. Figure 4.5 illustrates the series termination technique used to match a high impedance load. The value of the series resistor R_s is chosen such that R_s added to the source resistance R_T is equal to the characteristic impedance of the transmission line Z_o. The total resistance of the source added to the series termination, together with impedance of the transmission line, form a voltage divider. When a signal is launched, this voltage divider alters the signal such that

the amplitude is half its original value when it reaches the load. However, the load is assumed to possess a very high impedance therefore producing nearly 100 percent reflection. This reflection adds constructively at the load with the original signal to rebuild the full voltage level. The reflection then travels back to the source and is absorbed since the series termination added to the source forms a matched load [10].

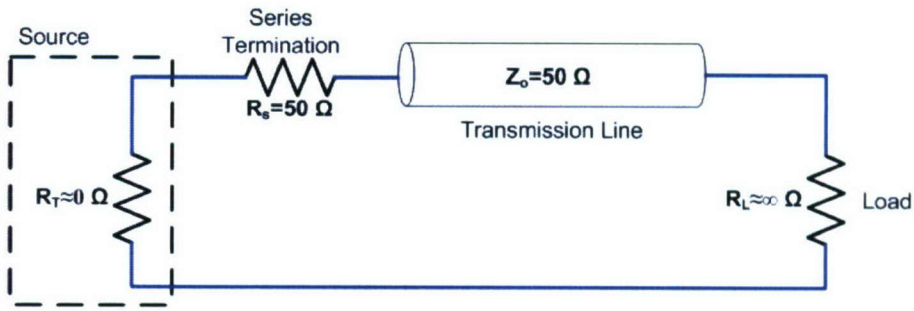


Figure 4.5: Series termination technique for preventing reflections on a transmission line as applied to the HSCD.

The series termination technique can be applied to the HSCD in the following manner. The 74LVTH245A octal bus transceivers have an output impedance which can be assumed to be much less than the 50-ohm impedance of the six feet of ribbon cable. In addition, the matching bus transceivers within the parallel-to-fiber optic converter have a high input impedance. For the series resistance calculation, R_T is assumed to be zero while R_L is assumed to be infinite. Therefore, the value of the series resistance is chosen to be 50 ohms. While these approximations may be overly simplified, a mismatch factor of 2 is usually satisfactory for acceptable signal integrity [11]. Therefore, the assumptions for R_T and R_L are reasonable for practical

purposes. The physical implementation of the series termination scheme in the HSCD will be discussed in Section 4.4.3.

4.4 Generating the Board Layout

The basic components of the HSCD, namely the connections to the radar, the FPGA, and the interface to the parallel-to-fiber optic converter, have been discussed in the previous sections. Due to complications at design time, the layout of the printed circuit board had to be divided into a main board and a daughter board. The main board is attached directly to the backplane of the radar and houses the FPGA and associated circuitry. The daughter board, whose primary function is to provide the interface circuitry for the parallel-to-fiber optic converter, is then attached to the main board.

The high speed data capture PCB design is limited to four layers since PCBs with additional layers beyond four are significantly more expensive to manufacture. However, a four-layer circuit board is more than adequate to provide proper signal routing and power distribution. In the following sections, the four-layer routing strategies employed for proper power distribution and noise immunity will be discussed.

4.4.1 Power Considerations and Noise Immunity

The primary feature of the four-layer PCB design is the ability to use two internal layers as dedicated power and ground supply planes while still having the two outer layers available for signal routing. One important purpose of dedicated power and ground planes is to supply integrated circuits mounted to the PCB with a low inductance path to the power source or bulk decoupling capacitors [12]. This is important

in situations where several of an IC's outputs may be switching simultaneously causing large fluctuations in the chip's input current. Without a low inductance path to the power source, noise is generated on the power rails as the voltage will drop temporarily whenever the supply current is not readily available. This power supply noise can then affect the operation of other chips connected to the same supply rails thereby drastically reducing the reliability of the PCB.

Dedicated power and ground planes also allow for controlling the impedances of critical traces. The return current for high frequency or fast rise-time signals tends to follow directly beneath the microstrip line. This occurs because the return current searches for the path of lowest impedance. Following directly beneath the microstrip line minimizes the loop area of the circuit thereby minimizing the overall inductance and therefore the impedance of the line. Solid power planes provide an uninterrupted return path such that the return current can consistently follow beneath the trace to maintain a constant impedance. This constant impedance minimizes reflections due to microstrip routing and ensures acceptable signal integrity.

4.4.2 Main Board Layout

The locations of the API and PCU connectors on the backplane of the radar are the determining factors for defining the dimensions of the HSCD main board. The PCB is designed so that it can be directly fitted onto the backplane of the radar without the need for extra cabling or connectors. Therefore, the PCB must be wide enough to reach both the PCU and API connectors shown in Figure 4.2. The usable dimensions of the PCB are therefore slightly larger than the dimensions called out in Figure 4.2.

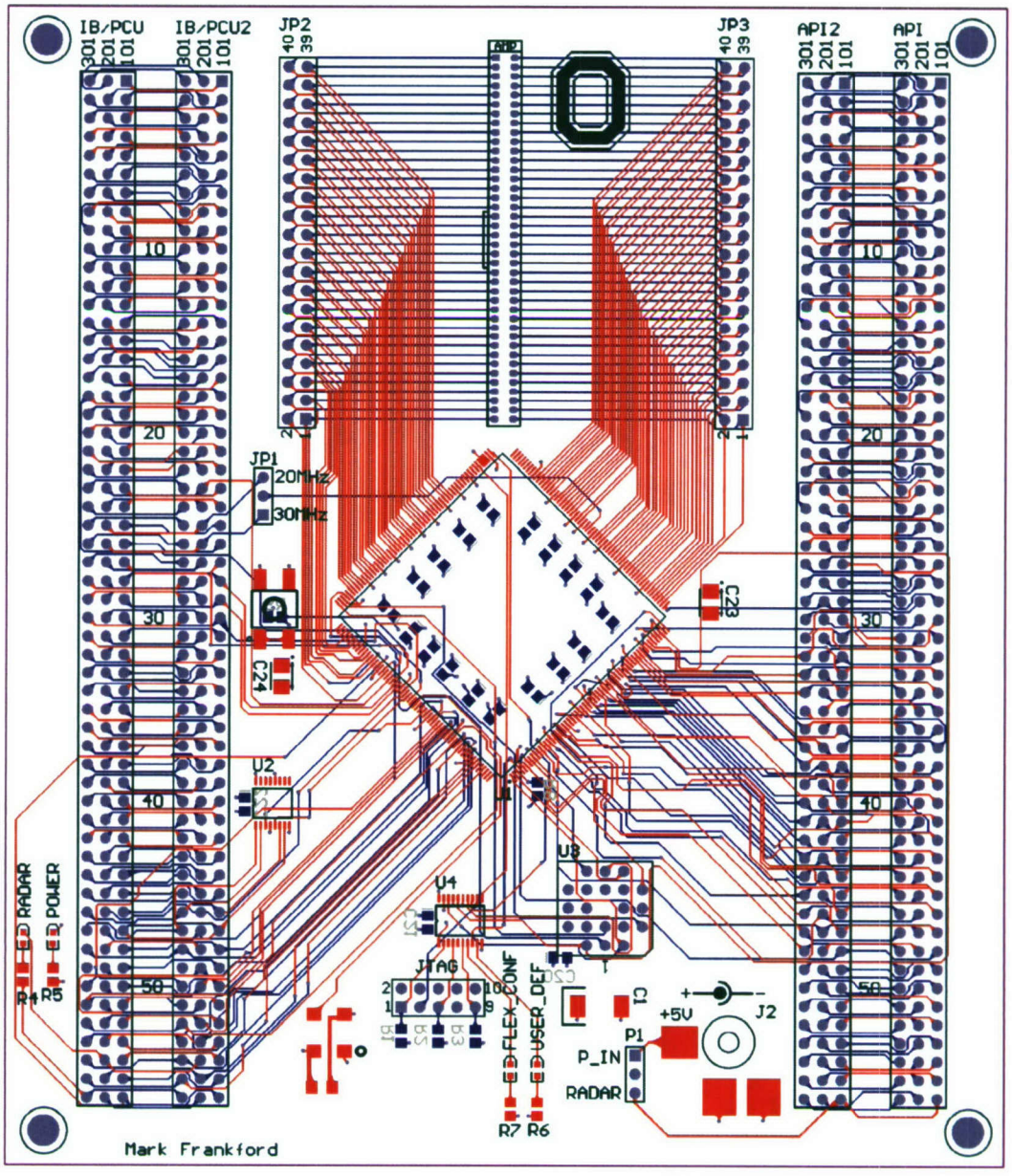


Figure 4.6: Main board layout showing top-layer components with traces (red) and bottom layer components with traces (blue).

Figure 4.6 shows the layout of the main board in its complete form. The red traces and red footprints are found on the top layer, while the blue traces and blue footprints are found on the bottom layer. The inner ground and power planes are hidden to simplify the view. The largest and most notable features on the board are the connections to the backplane of the radar and the identical test headers. These are labeled on the layout as IB/PCU on the left which connects to the pulse compression unit, and API on the right which connects to the antenna processor interface. The identical test headers, which directly pass the signals from the backplane to male headers on the top of the board, provide a way of monitoring the radar's operation while the HSCD is in place on the backplane since the original backplane pins are unaccessible. This is an important feature and is extremely useful in debugging algorithms running in the FPGA.

In addition to the backplane connections, the main board houses the FPGA and its associated programming hardware including the EPC2 serial flash memory and JTAG port. The main board also houses a 40 MHz crystal oscillator whose output is supplied to the FPGA. An additional clock source is supplied by a jumper which connects either the radar's 20 MHz or 30 MHz clock signals to the FPGA. These clock sources are used to drive the FPGA software designs which will be discussed in Chapter 5.

A power jack is provided to connect a DC adapter to the HSCD when power is not available from the radar. Manual selection between the radar's power supply or the external DC adapter is accomplished by setting a jumper. This ensures that the external power supply and the radar's power supply are never shorted together. Two LEDs on the lower left of the PCB indicate when power is available from the

radar's backplane and when the HSCD is currently powered on. Two additional LEDs located at the bottom center of the PCB provide further visual diagnostics. The LED labeled *FLEX_CONF* is only illuminated when the FPGA programming process has been completed. Once the FPGA has been successfully programmed, the code has access to the *USER_DEF* LED which can indicate if the system is running properly.

The FPGA footprint was placed with a 45-degree rotation with respect to the other components on the main board to facilitate simpler routing. The connections to the radar's backplane for both the API and PCU are located toward the bottom of the PCB such that they can be easily routed to the two sides of the FPGA pointing downwards. The remaining two sides of the FPGA are used for the outputs required to drive the interface to the parallel-to-fiber optic converter. These output signals are routed to headers, labeled JP2 and JP3. If the FPGA is not placed with a 45-degree rotation, signals from at least one side of the FPGA would have been routed beneath the FPGA to reach their destinations. This would increase the total length of the traces as well as the total number of vias on the board.

The headers, JP2 and JP3, were originally designed into the board as a way to debug the output of the FPGA while the parallel-to-fiber optic converter was connected to the AMP® System 50 connector located at the top center of the main board. However, it was discovered after further communication with the manufacturer of the parallel-to-fiber optic converter that the 74LVTH245A 3.3V octal bus transceivers mentioned earlier would be required. Due to time constraints, it was decided that a separate daughter board would be built to house these bus transceivers. This daughter board directly connects to JP2 and JP3 to gain access to the outputs of the FPGA and then send the signals to the parallel-to-fiber optic converter

through its own AMP® System 50 connector. The design of this daughter board will be discussed in the following section.

4.4.3 Daughter Board Layout

The HSCD daughter board connects to headers JP2 and JP3 on the main board to route the outputs from the FPGA to the 74LVTH245A 3.3V octal bus transceivers and eventually to an AMP® System 50 connector and ribbon cable according to the pin assignments given in Table 4.4. The final layout of the daughter board is given in Figure 4.7. The AMP® System 50 connector can be clearly seen at the top center of the layout. The 74LVTH245A bus transceivers are denoted by the designators U1, U3, U4, U5, U6, and U7. The inputs to these transceivers are connected to headers P1 and P2 which connect to headers on the main board JP2 and JP3, respectively. The outputs of the transceivers are then connected to the AMP® System 50 connector through 50-ohm series termination resistors discussed in Section 4.3.

The daughter board shares its 4-layer design with the main board which consists of top and bottom signal layers with internal ground and power planes. The thickness of these layers is not important in the main board layout, but plays a key role in the layout of the daughter board. Figure 4.8 shows a side view of the layer stack-up. The thickness of each layer along with the dielectric constant of the FR4 material was obtained from the printed circuit board manufacturer. This information is used to calculate the width of the microstrip line required to produce a given characteristic impedance. The microstrip traces running from the series termination resistors to the AMP® System 50 connector are designed with a characteristic impedance of 50 ohms. This requires a width of approximately 16 mils in order to minimize impedance

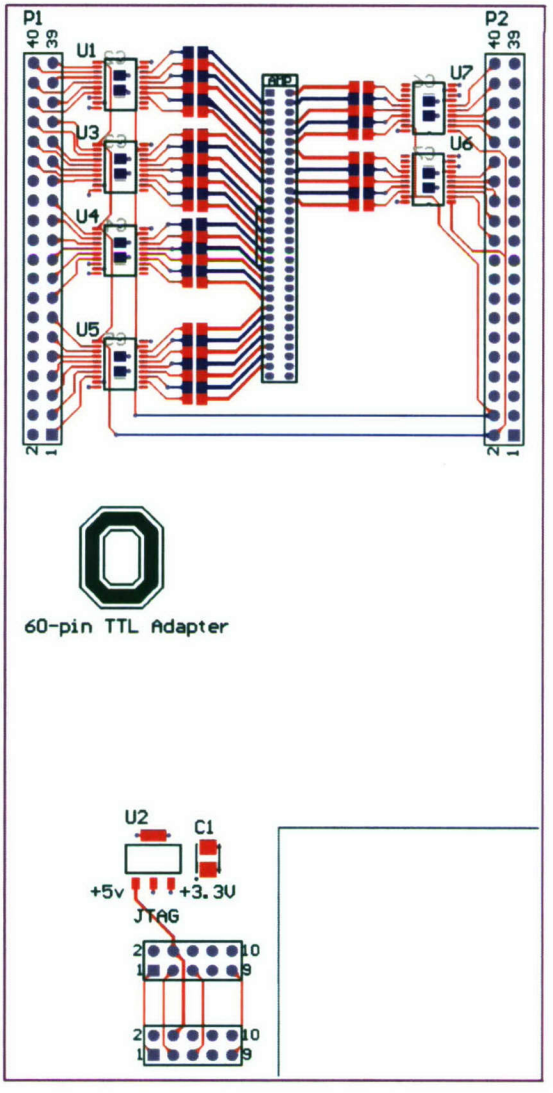


Figure 4.7: Daughter board layout showing top-layer components with traces (red) and bottom layer components with traces (blue).

discontinuities at the transition between the daughter board and the 50-ohm ribbon cable. This ensures minimal reflections at the connector thereby helping to maintain acceptable signal integrity.

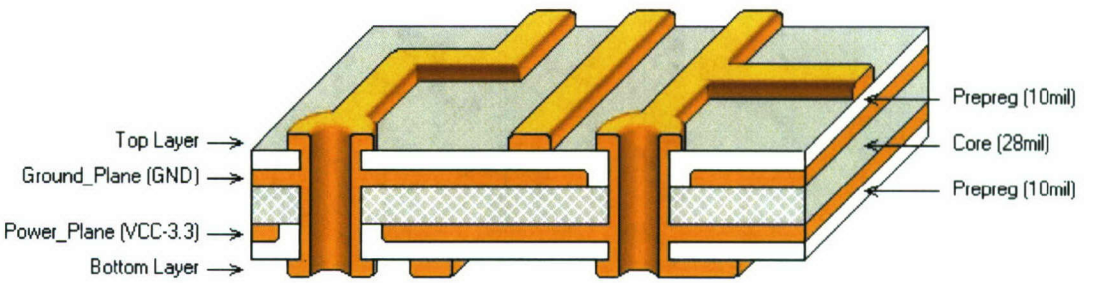


Figure 4.8: Daughter board layer stack-up showing the thickness of the dielectric between each layer.

Since the 74LVTH245A bus transceivers require a supply voltage of 3.3V, a voltage regulator capable of converting the 5V from the main board is needed. However, the 5V supply is available on neither the JP2 nor the JP3 header. Therefore, the daughter board is long enough to connect to the JTAG header on the main board which includes a 5V connection and a ground connection. A low-dropout voltage regulator, labeled U2 in Figure 4.7, converts this 5V supply into 3.3V to power the bus transceivers.

Since it would be advantageous to still have access to the JTAG port even with the daughter board in place, the JTAG port is mirrored on the daughter board. The lower header on the daughter board connects to the JTAG port on the main board below it to gain access to the power supply, while the upper header provides access to the JTAG port from above.

4.5 Final Board Assembly

The HSCD main board and daughter board layouts were sent to an outside company to be manufactured on standard Flame Resistant 4 (FR-4) PCB material. Upon arrival, they were carefully populated by hand and tested for assembly errors where necessary. The FPGA, EPC2, and JTAG header on the main board were assembled first to allow for programming and testing of the FPGA, as this was the most complex portion of the design. Once this was completed, the rest of the main board and the daughter board was assembled.

Figure 4.9 is a photograph of a partially assembled prototype main board which shows the test connectors used for monitoring the signals from the API and PCU while the HSCD is attached to the radar. The two headers where the daughter board will be be placed are also visible. Figure 4.10 is a photograph taken of the bottom of the same prototype main board revealing the two large connectors which will be pressed onto the radar's development chassis to connect to the PCU and API as described in Figure 4.2.

Photos of the completely assembled HSCD are shown in Figures 4.11 and 4.12. In these figures, the daughter board is attached to the main board as previously described. Two large handles have been added to the assembly to aid in attaching and removing the HSCD from the radar's backplane since a significant amount of force is required. The AMP® System 50 connector in the center of the daughter board where the parallel cable running to the parallel-to-fiber optic converter will attach is clearly visible in Figure 4.11.

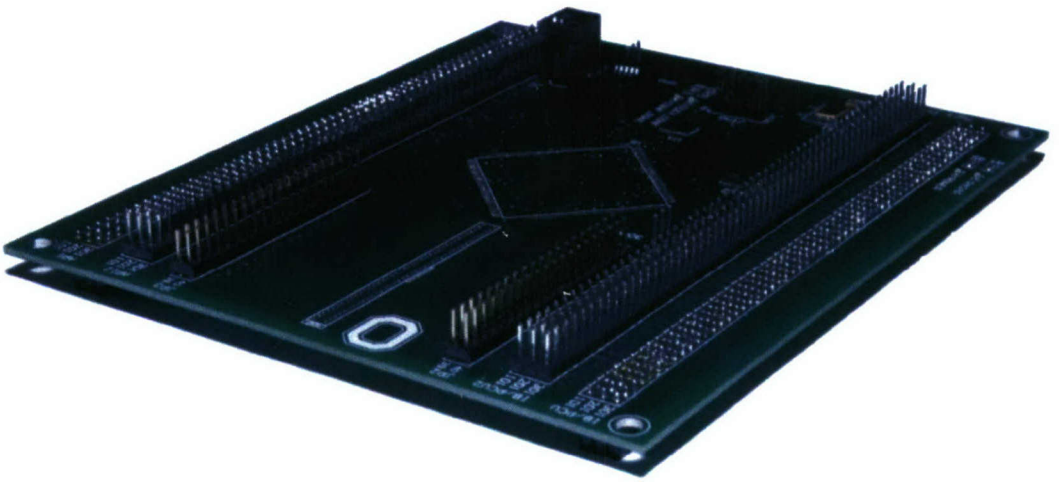


Figure 4.9: Top view of a partially assembled main board prototype showing the API and PCU test headers accessible from the top, as well as the JP2 and JP3 headers where the daughter board is attached.

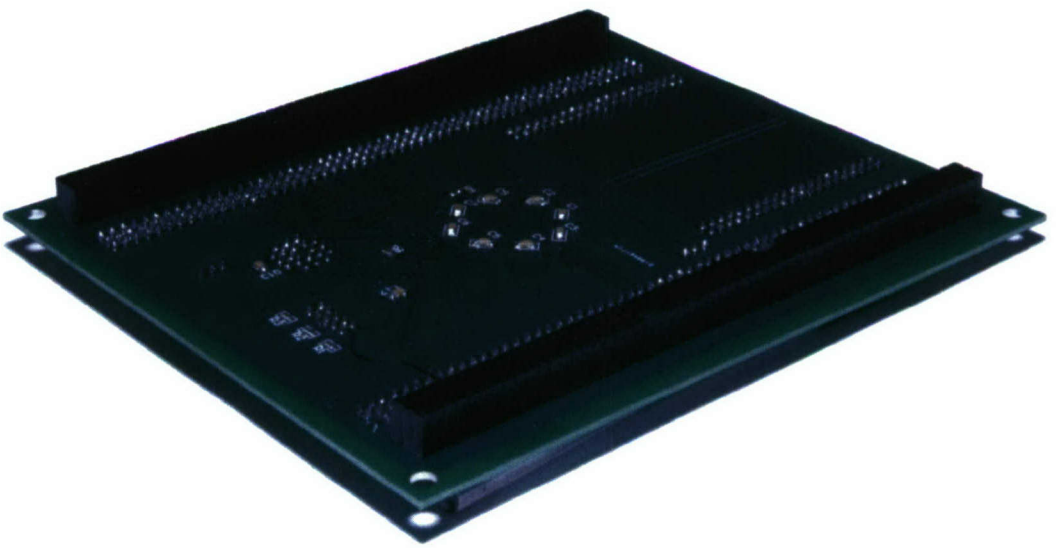


Figure 4.10: Bottom view of a partially assembled main board prototype showing the two sets of black female headers where the main board will connect to the API and PCU on the special radar backplane.

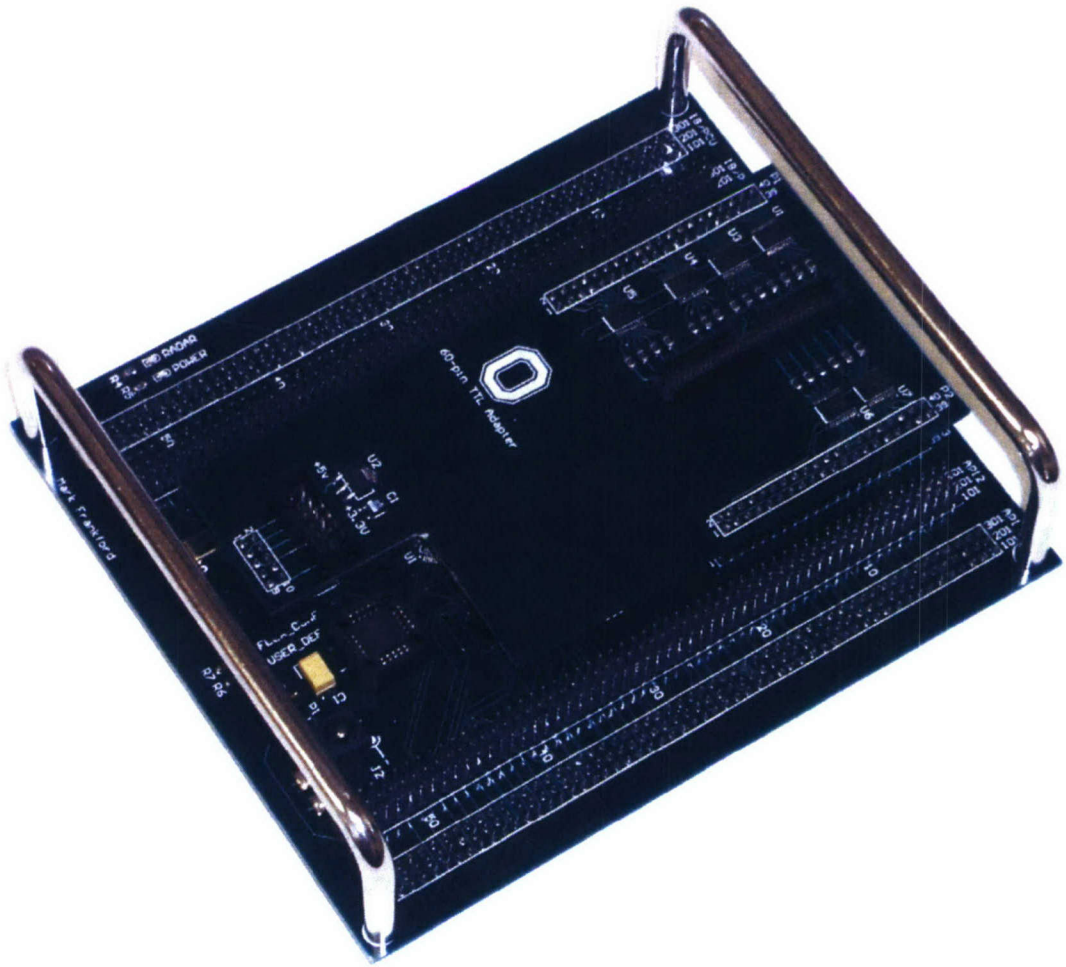


Figure 4.11: Top view of the completely assembled HSCD main board with daughter board attached.

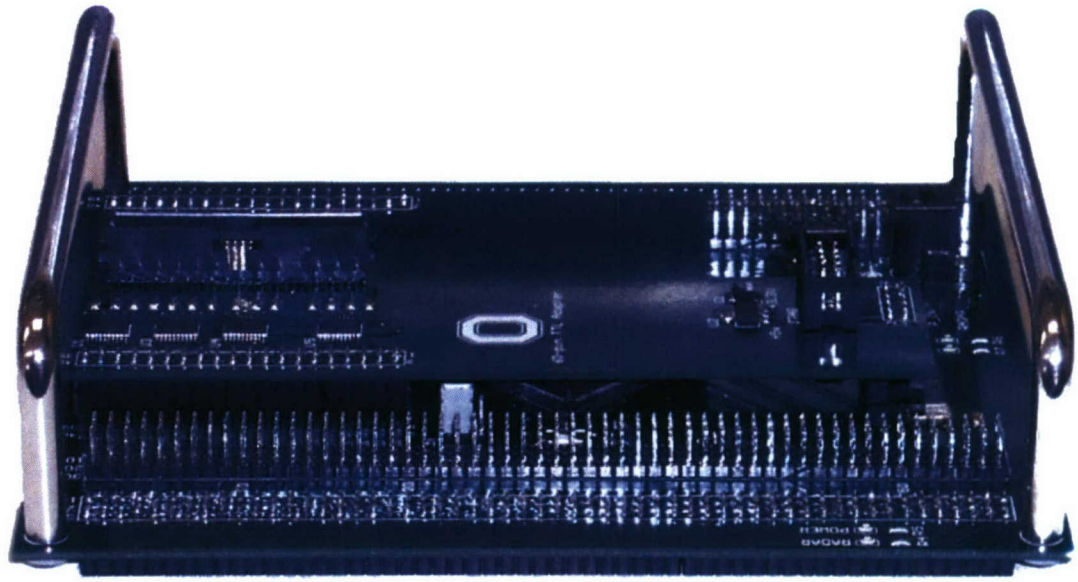


Figure 4.12: Side view of the completely assembled HSCD main board with daughter board attached.

4.6 Conclusion

This chapter has examined the physical design of the HSCD PCB. The design relies on the flexibility and configurability of the Altera Flex10K® FPGA. The following chapter will examine the software which will enable the Flex10K® to achieve the goals of the HSCD.

CHAPTER 5

SOFTWARE IMPLEMENTATION

The previous chapter described the HSCD hardware in detail, carefully examining each component required to properly transfer data from the radar hardware to the parallel-to-fiber optic converter and eventually to the data recording PC. This chapter will describe the software necessary for the HSCD to function correctly in this role, written in Very High Speed Integrated Circuit (VHSIC) Hardware Description Language (VHDL). Before delving directly into the software, the information which the HSCD is required to produce is first examined. Once this has been described, an output data format will be established which dictates how the information is packaged for transfer to the parallel-to-fiber optic converter. The software which is written to adhere to this standard will then be examined in detail. Finally, the methods used to test the functionality of the programmed HSCD will be shown.

5.1 Required Data

The HSCD was devised as a way to record the raw in-phase and quadrature data samples from the radar's receivers and synchronize those samples with the position of the antenna. Therefore, the primary data required consists of the 16-bit in-phase and quadrature samples, or range gates, and all data pertaining to the antenna's

requested and measured position and velocity. However, to perform any meaningful processing on this data, additional information must be gathered.

The antenna position and velocity data transfers to and from the API occur far less frequently than the in-phase and quadrature data transfers to the PCU. Therefore, it is important to tag the captured data from both the API and PCU with a value which identifies in which CI the data transfer occurred. Furthermore, the in-phase and quadrature data samples from the PCU must be tagged with a value which identifies which IPP within the CI the data transfer occurred. Also, since every RG within an IPP consists of one in-phase data sample and one quadrature data sample, each sample must be tagged with a value to identify the RG number so they can be reunited to form an in-phase/quadrature pair later during analysis. Finally, a running time stamp is bundled with all captured data to provide an accurate time scale during post-processing. The next section will discuss how this information is organized for transfer to the parallel-to-fiber optic converter.

5.2 Output Data Format

The information generated by the HSCD must be packaged into a format readily transferable over the parallel cable discussed in the previous sections. Since the information will be reformatted by an FPGA residing within the parallel-to-fiber optic converter before being sent to the data recorder computer, this data structure should be fairly straightforward and easily interpreted. Therefore, a standard has been devised which utilizes the 36 signal lines available on the parallel cable.

The devised standard generates two 35-bit data words for every RG sample or API parameter. The 36^{th} bit available on the ribbon cable is used as a data strobe

to clock the data words into the FPGA in the parallel-to-fiber optic converter. The 35^{th} bit on the ribbon cable is used to identify the word as either belonging to a RG word set or an API word set, while the 34^{th} bit is used to identify whether it is the first or second word of the set. Since the arrivals of data from the API and PCU are independent and can happen in any order, it is possible for the first and second words of a RG word set to be interleaved with the first and second words of an API word set. However, two RG word sets will never be interleaved, nor will two API word sets be interleaved. For example, if the first word in a RG word set is received, the next word will either be the second word in the set, or the first word in an API word set. It will *never* be the first word in a new RG word set. This eliminates any ambiguity on the receiving end as to how the incoming data words should be arranged.

Appendix A describes the pin mapping between the software-defined labels in VHDL, the FPGA pins, and the AMP® System 50 connector pins. The software-defined pin labels start at 35 and countdown to 0, so the 36th signal in the ribbon cable is connected to `to_connector[35]` and while the 1st signal is connected to `to_connector[0]`. The following sections will discuss the rest of the signals not yet covered and how they are organized into RG word sets and API word sets.

5.2.1 Range Gate Data Word Sets

As previously mentioned, the 36^{th} through 34^{th} signals of the ribbon cable are used as a write strobe, RG or API word set selector, and first/second word identifier respectively. This leaves the 33^{rd} through the 1^{st} open for transferring actual data. Table B.1 of Appendix B describes how the data is arranged for RG data word sets.

The first column of these tables is the software-defined label which can be cross-referenced in Appendix A to find the corresponding ribbon cable pin assignment. The second column in these tables shows the type of data assigned to the pins during the first RG data word, while the third column shows the type of data assigned during the second RG data word. The first RG data word contains the 16-bit value of the in-phase or quadrature RG sample captured from the PCU along with a 16-bit running time stamp. The second RG data word contains a bit which identifies the contents of the RG data set as either in-phase or quadrature. In addition, the second RG data word also contains a bit which identifies the RG data as originating from channel A or channel B and a status bit indicating whether or not the integrator in the receiver has been saturated. Three 9-bit numbers identifying the RG, IPP, and CI in which the data was captured also reside in the second RG data word.

5.2.2 Antenna Interface Word Sets

The API data word sets share a similar format to the RG data word sets previously introduced. Table B.2 of Appendix B describes how the data is arranged for API data word sets. As can be expected, the 36^{th} through the 34^{th} bits share the same functionality as the RG data word sets. The first API data word contains a 16-bit value representing some antenna parameter. The first API data word also contains a 16-bit running time stamp generated by the same source as the time stamp in the RG data word sets. The second API data word contains an 8-bit key which identifies what type of API parameter is being sent in the first data word. This 8-bit key is simply the lower eight bits of the bus address discussed in Section 3.3.3. Additionally, the second API data word contains a 9-bit number identifying the CI in which the

data was captured. This CI number will be used to correlate the antenna parameters with the RG data tagged with an identical CI number.

5.2.3 A Few Notes about RG, IPP, and CI Labels

The RG, IPP, and CI identifying numbers have a few important properties. Firstly, the RG number is incremented for every range gate within a given IPP. The first RG is labeled zero with that number being incremented for each successive range gate. However, at the end of an IPP, the RG label is reset to zero. The labeling for the IPP within a CI is slightly different due to features of the radar's hardware. The first IPP of a CI is labeled with a value of 1 with the caveat that no data is transfered from the receivers to the PCU during the first IPP since data is still being buffered in the receivers. Therefore, the first stream of data intercepted by the HSCD occurs in the second IPP, causing the label to be equal to 2. The IPP labels are set back to 1 at the end of a CI just as the RG labels are set back to 0 at the beginning of an IPP. The CI labels are incremented at the beginning of every CI. However, there is currently no way to discern when the radar has completed scanning an entire bar with the current HSCD hardware, so the CI labels reach a maximum value of $2^9 - 1$ and simply roll over back to 0. This is not a problem since the data is correlated with the antenna position, and the completion of a bar is easily discerned from this data.

5.3 Software Design for Capturing and Buffering Data

The FPGA software is written to capture the in-phase and quadrature RG samples and API data while simultaneously keeping track of the current RG count, IPP count, and CI count. The in-phase and quadrature data and the API data are combined with the proper RG, IPP, and CI tags to form the RG data word sets and API data

word sets described earlier. The software buffers these word sets as necessary in first-in-first-out (FIFO) memories before they are interleaved intelligently according to the predetermined standard described in Section 5.2. This whole process can be highly parallelized within the FPGA, with each step being performed by specific functional blocks. Figure 5.1 shows a high level block diagram generated with the Altera Quartus® II design software. Each block in the diagram represents a function or state machine written in VHDL which performs a specific task. These blocks are connected together using lines which represent wires and buses. Colored squares have been added to partition the diagram into smaller pieces which will each be discussed in turn in the following sections.

5.3.1 CI and IPP Capture

The green section in Figure 5.1 contains the functionality necessary to properly track both the current CI and the current IPP within the CI. The block takes four inputs from the radar which signal the first and last IPPs of a CI as well as the start and end of each IPP within a CI. The functional blocks labeled `IPP Counter Control` and `CI Counter Control` use these inputs to increment two 9-bit counters at the end of each IPP or CI respectively. The `IPP Counter Control` function also properly resets the IPP counter at the end of each CI such that the first IPP in the next CI is properly tagged as previously noted in Section 5.2.3. The outputs of the green block are two 9-bit values which always represent the current IPP count and CI count at any given moment in time. These outputs are then available to the rest of the design for bundling with other data.

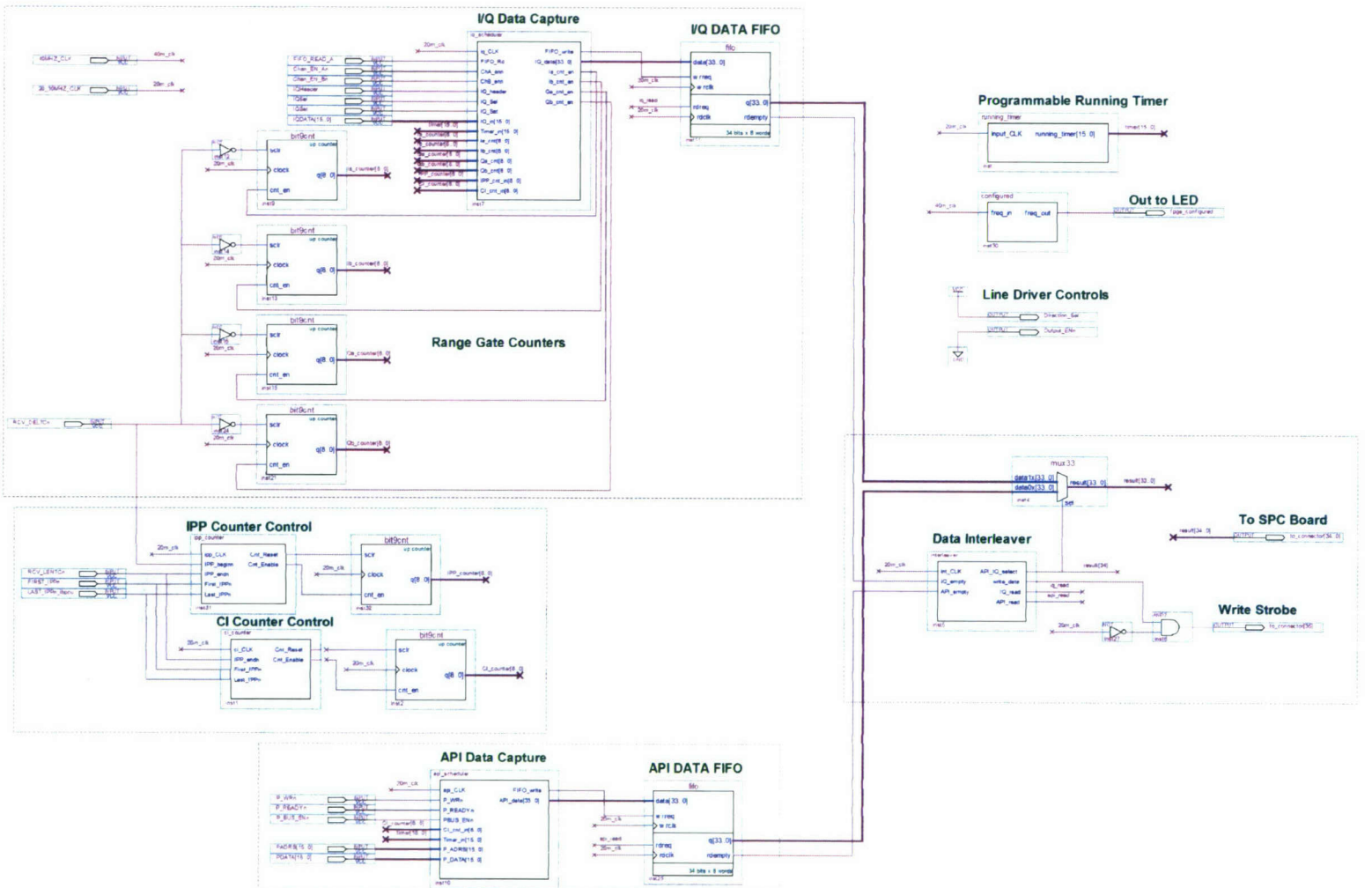


Figure 5.1: Graphical VHDL layout in Quartus® with the four major functional sections color-coded.

5.3.2 In-phase and Quadrature Data Capture

When any RG data transfer occurs between the radar receivers and the PCU, the IPP count and CI count data discussed in the previous section are combined with a RG count and the in-phase/quadrature RG data generated by the red section in Figure 5.1. Since the in-phase samples are transferred separately from the quadrature samples for each receiver, separate counters are required to keep track of the RG count within an IPP for each in-phase or quadrature channel from receiver A or receiver B. The `I/Q Data Capture` function controls each of the four counters utilizing the signals available from the radar listed in Table 4.1 previously discussed. These counter values, along with the CI and IPP counter values from the previous section are combined within the `I/Q Data Capture` function with the actual 16-bit RG data and a 16-bit time stamp to form a RG data word set. The two words in this set, which adhere to the output data format outlined in Section 5.2, are then pushed into a FIFO memory which will eventually be serviced by the blue section in Figure 5.1 discussed later. This process occurs for every in-phase or quadrature sample sent from either receiver to the PCU.

5.3.3 Antenna Interface Data Capture

The API data capture functionality is carried out by yellow section in Figure 5.1. The inputs to this section are those signals listed in Table 4.2 along with the CI count previously discussed. The block labeled `API Data Capture` is designed to listen to the bus traffic between the radar's central processor and the various subsystems connected to the bus described in Section 3.3.3. Any data transfer involving the API is easily distinguishable by a hexadecimal value of $10XX in the ADRS (address) signal. The

least significant eight bits of the address identify what type of data is being transfered to or from the API and are defined by the radar's software. Therefore, the `API Data Capture` function simply records the 16-bit bus data along with the least significant eight bits of the address whenever a bus transfer occurs with an address equal to $10XX. These two pieces of information are then bundled with the CI count and a 16-bit time stamp into two words to form an API data word set adhering to the output format outlined in Section 5.2. This two word set is then pushed into a FIFO memory, separate from the RG data FIFO memory mentioned previously, which will eventually be serviced by the blue section in Figure 5.1.

5.3.4 Buffering and Interleaving

The previous two sections ended with RG data word sets and API data word sets each being pushed into their own respective FIFO memories. Due to limitations with the FPGA, the size of these FIFO memories is constricted such that they are only eight addresses deep. Therefore only four complete data word sets can be stored in one of these FIFO memories before data corruption occurs. The data interleaving functionality contained in the blue section of Figure 5.1 ensures that neither FIFO overflows its capacity and that the data interleaving standard discussed in Section 5.2 is adhered to. The `Data Interleaver` function achieves this by monitoring FIFO memory status bits which indicate when a FIFO is empty and acting accordingly. The `Data Interleaver` function utilizes a multiplexer to selectively connect the output of either of the two FIFO memories to the FPGA's output pins labeled `to_connector[33]` down to `to_connector[0]`. The multiplexer control signal which

switches which FIFO is connected to the output is also an output itself, as it becomes the RG or API word select bit as defined in Appendix B as `to_connector[34]`.

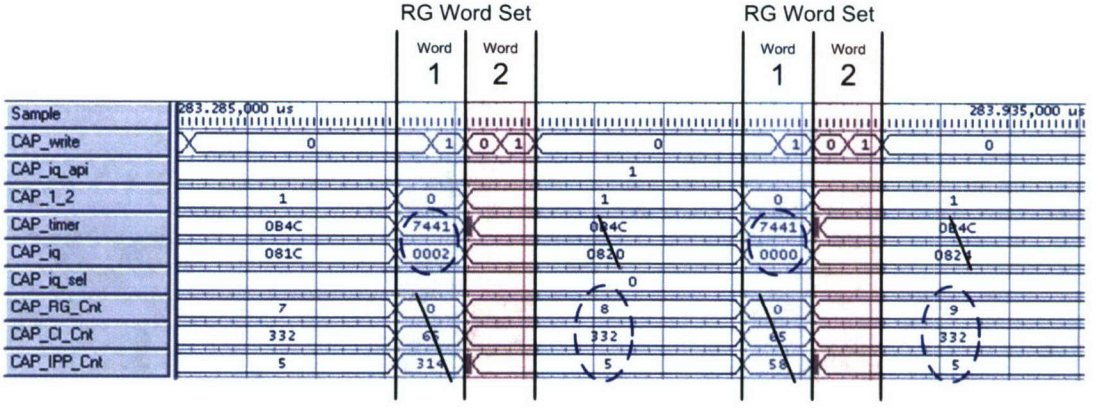
To ensure that neither the RG nor API FIFO overflows, only a single word is transferred out of either FIFO at a time. For instance, if the RG FIFO is not empty, a single word will be read and sent over the ribbon cable. Otherwise, no data is sent from the RG FIFO, and the API FIFO status bit is then checked. Similarly, if the API FIFO is not empty, a single word is read from this FIFO and sent over the ribbon cable. Otherwise, no data is sent from the API FIFO, and the RG FIFO is checked again. In this manner, both FIFO memories are given an equal opportunity to be read. This checking process occurs fast enough such that the contents of the FIFOs are never corrupted.

5.4 Testing and Verification

To verify the proper functionality of the software, the HSCD main board is connected to the radar's backplane. A logic analyzer is used to monitor the test headers which provide identical connections to the radar's backplane as if the HSCD were not present. Without the daughter board in place, headers JP2 and JP3 are accessible and can be monitored with the logic analyzer as well. In this way, data flowing into the FPGA from the radar and data flowing out of the FPGA to the parallel-to-fiber optic converter can be monitored. Verification that the data flowing into the FPGA is properly handled to create well-formed output data is therefore a simple task.

Figure 5.2 shows a screen shot of output data captured by the logic analyzer. A table is included which lists the signal names displayed with the function each represents. The 36-bit output data word has been broken up into its various pieces,

with parts of the lower 32 bits of the word being assigned to more than one label by the logic analyzer to make the data easier to interpret. The valid data in either the first or second word of the data word set has been circled, with a line being drawn through the nonsensical data. The figure shows two complete RG word sets in the previously described output data format. Close examination shows that these RG word sets contain the 8^{th} and 9^{th} in-phase RG samples in the 5^{th} IPP of the 332^{nd} CI.



Label:	Fuction:
CAP_Write	Write strobe (Data is valid on the rising edge.)
CAP_iq_api	RG or API data select
CAP_1_2	1st or 2nd word select
CAP_timer	Running time stamp
CAP_iq	RG data
CAP_iq_sel	In-phase or quadrature select
CAP_RG_Cnt	RG count value
CAP_CI_Cnt	CI count value
CAP_IPP_Cnt	IPP count value

Figure 5.2: Screen shot of actual data in the proper output data format captured using a logic analyzer.

It is easily verified that the data sent into the FPGA is being properly formatted before being sent to the parallel-to-fiber optic converter. However, it is necessary to ensure that the in-phase and quadrature RG data being sent into the FPGA is in fact comprised of valid RG data samples from the radar's receivers. To accomplish this, the logic analyzer is set up to record all of the RG data words for all IPPs within a given CI. This data is written to a tab-delimited text file so that it can be imported into MATLAB for processing. A script was written in MATLAB which imports this data and assembles the in-phase and quadrature data into valid complex-valued data pairs. The script then plots the Fast Fourier Transform (FFT) of the in-phase/quadrature data pairs to analyze the data in the frequency domain. The plot of this FFT is shown in Figure 5.3. It must be noted that there was no transmitter in the radar when this data was recorded so the sampled RF signal consisted only of noise and leakage from other RF sources. However, hardware within the receivers was switching this leakage on and off which effectively produced a pulsed waveform with a very small amplitude. The figure clearly shows the spectrum of a pulsed waveform, and the similarity between Figure 5.3 and Figure 2.5(b) is evident.

5.5 Conclusion

This chapter began by examining the types of data the HSCD is required to produce. A practical output data format was then described which established a formal outline of how the FPGA should handle the data. A VHDL software design was then graphically examined at a high level to show how the data capture device actually intercepts data and adheres to the output data format when sending this

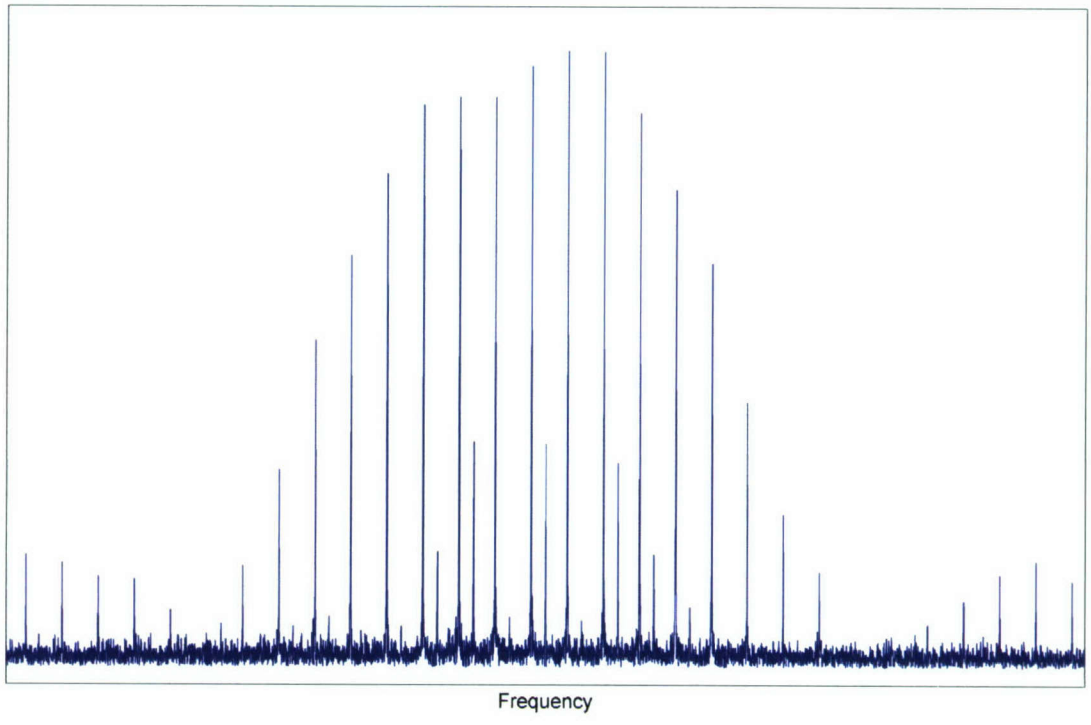


Figure 5.3: Fast Fourier Transform (FFT) of RG data recorded by the HSCD.

data to the parallel-to-fiber optic converter. Finally, captured data was analyzed to ensure that the whole system functions correctly.

CHAPTER 6

CONCLUSION

6.1 Conclusion

This thesis has described the design, assembly, and programming of a HSCD utilized in the recording of range samples and antenna positions from a given pulse doppler radar contained in a coherent RGASM test bed. This thesis first discussed the general features of coherent RGASMs, specifically the pulse doppler radar and those features which distinguish it from non-coherent radars. A coherent receiver was then examined to gain general insight into what types of data are generated by the receiver and which types may be of interest. Following this general RGASM overview, specific components of the actual radar used by the RGASM test bed were introduced including the antenna processor interface (API) and pulse compression unit (PCU). Special attention was given to the radar's scanning behavior and special backplane architecture so that a strategy could be proposed for capturing the required information. The HSCD hardware was then described followed by an overview of the software which controls the device.

The HSCD has been built, programmed, and tested using the actual radar hardware with which it will be used in the RGASM test bed. Complete functionality has

been demonstrated by capturing the data flowing into and out of the HSCD using a logic analyzer and closely examining it to ensure that no data is lost or corrupted and that the output data format is implemented correctly. With the completion of these tests, the HSCD is ready to function as an integral part of the coherent RGASM test bed.

6.2 Future Work

While the HSCD fulfills the original set of requirements placed upon it, there are a few changes which could be made to improve upon its design. The first and most obvious change would be to integrate all daughter board functionality directly into the main board. This would create a more sturdy, compact, and reliable design than the current two-board scheme.

In addition to this simple layout change, two improvements could be made which enhance the functionality of the HSCD. The first improvement would be to add the capability to record data from an additional data source called the high-speed bus, or HBUS. The second improvement would be to integrate the fiber-optic data link directly onto the HSCD therefore bypassing the current parallel-to-fiber optic converter. These potential modifications are discussed in the remaining to sections.

6.2.1 High-Speed Bus Capture Implementation

The HSCD currently captures in-phase and quadrature range samples, the antenna's position, and antenna status words from buses traveling over the backplane of the radar. There is a third bus present on the radar's backplane which carries additional radar operational parameters which are of interest. This bus is called the

high-speed bus, or HBUS. Data flowing over this bus is significantly more complex and includes DSP processing parameters and processing results.

In order to capture this data, significant changes would need to be made to the HSCD. Access to the HBUS is already available through the PCU connector so additional connections could be made to the FPGA. However, the FPGA itself would be stretched to its limits with the addition of such an active data source. Therefore, a larger and faster FPGA would be required to handle this new influx of data. In addition, the output data format would have to be expanded to include this new data set and would most likely require significant changes. Even so, having HBUS data available while trying to examine the operation of an RGASM would be extremely desirable. This makes pursuit of HBUS functionality very worthwhile.

6.2.2 Fiber-optic Transceiver Integration

Currently, the complete data recording system consists of those components outlined in Figure 4.1. The VME-based parallel-to-fiber optic converter contains an extremely capable FPGA connected to a VME fiber-optic transceiver card. This FPGA was originally intended to be used for preprocessing the data before it is sent to the data recording computer. However, it was decided that not much preprocessing would be utilized. Therefore it would be advantageous if a fiber-optic transceiver could be integrated directly onto the HSCD, thereby wholly eliminating the VME-based parallel-to-fiber optic converter. Several fiber-optic transceiver solutions where examined which could be placed on a custom PCB. Using a more modern FPGA than the Flex10K® would ensure that data could be processed quickly enough to utilize the available bandwidth afforded by these fiber-optic transceiver solutions. Overall,

it was determined that integrating a fiber-optic transceiver onto the HSCD is feasible and could be successfully pursued in the future.

APPENDIX A

OUTPUT PIN MAPPING BETWEEN AMP® CONNECTOR, FPGA, AND SOFTWARE

AMP® Pin	FPGA Pin	Software Design Name
2	111	to_connector [35]
3	137	to_connector [34]
4	109	to_connector [33]
5	138	to_connector [32]
7	139	to_connector [31]
8	107	to_connector [30]
9	141	to_connector [29]
10	105	to_connector [28]
12	102	to_connector [27]
13	142	to_connector [26]
14	100	to_connector [25]
15	143	to_connector [24]
17	144	to_connector [23]
18	98	to_connector [22]
19	146	to_connector [21]
20	95	to_connector [20]
22	88	to_connector [19]
23	147	to_connector [18]

Continued

Table A.1: Output data map showing the relationship between the AMP® System 50 connector pins, Altera Flex10K® FPGA pins, and software VHDL design names.

Table A.1 continued

24	86	to_connector [17]
25	148	to_connector [16]
27	149	to_connector [15]
29	151	to_connector [14]
31	152	to_connector [13]
33	154	to_connector [12]
35	157	to_connector [11]
37	159	to_connector [10]
39	162	to_connector [9]
41	164	to_connector [8]
43	167	to_connector [7]
45	169	to_connector [6]
47	172	to_connector [5]
49	174	to_connector [4]
51	184	to_connector [3]
53	191	to_connector [2]
55	193	to_connector [1]
57	194	to_connector [0]

APPENDIX B

OUPUT DATA FORMAT

Design Name:	**RG Data Word #1**	**RG Data Word #2**
to_connector [35]	Write Strobe	Write Strobe
to_connector [34]	RG or API Data Select = 1	RG or API Data Select = 1
to_connector [33]	Word Identifier = 0	Word Identifier = 1
to_connector [32]	UNUSED	UNUSED
to_connector [31]	Range Gate I/Q Data [15]	UNUSED
to_connector [30]	Range Gate I/Q Data [14]	UNUSED
to_connector [29]	Range Gate I/Q Data [13]	I/Q Select (0=I, 1=Q)
to_connector [28]	Range Gate I/Q Data [12]	Receiver Saturated
to_connector [27]	Range Gate I/Q Data [11]	Chan. Sel. (1=Sum,0=Diff)
to_connector [26]	Range Gate I/Q Data [10]	Range Gate Number [8]
to_connector [25]	Range Gate I/Q Data [9]	Range Gate Number [7]
to_connector [24]	Range Gate I/Q Data [8]	Range Gate Number [6]
to_connector [23]	Range Gate I/Q Data [7]	Range Gate Number [5]
to_connector [22]	Range Gate I/Q Data [6]	Range Gate Number [4]
to_connector [21]	Range Gate I/Q Data [5]	Range Gate Number [3]
to_connector [20]	Range Gate I/Q Data [4]	Range Gate Number [2]
to_connector [19]	Range Gate I/Q Data [3]	Range Gate Number [1]
to_connector [18]	Range Gate I/Q Data [2]	Range Gate Number [0]
to_connector [17]	Range Gate I/Q Data [1]	IPP Number [8]
to_connector [16]	Range Gate I/Q Data [0]	IPP Number [7]
to_connector [15]	Running Time Stamp [15]	IPP Number [6]

Continued

Table B.1: In-phase and Quadrature output data word format.

Table B.1 continued

to_connector [14]	Running Time Stamp [14]	IPP Number [5]
to_connector [13]	Running Time Stamp [13]	IPP Number [4]
to_connector [12]	Running Time Stamp [12]	IPP Number [3]
to_connector [11]	Running Time Stamp [11]	IPP Number [2]
to_connector [10]	Running Time Stamp [10]	IPP Number [1]
to_connector [9]	Running Time Stamp [9]	IPP Number [0]
to_connector [8]	Running Time Stamp [8]	CI Number [8]
to_connector [7]	Running Time Stamp [7]	CI Number [7]
to_connector [6]	Running Time Stamp [6]	CI Number [6]
to_connector [5]	Running Time Stamp [5]	CI Number [5]
to_connector [4]	Running Time Stamp [4]	CI Number [4]
to_connector [3]	Running Time Stamp [3]	CI Number [3]
to_connector [2]	Running Time Stamp [2]	CI Number [2]
to_connector [1]	Running Time Stamp [1]	CI Number [1]
to_connector [0]	Running Time Stamp [0]	CI Number [0]

Design Name:	API Data Word #1	API Data Word #2
to_connector [35]	Write Strobe	Write Strobe
to_connector [34]	I/Q or API Data Select = 0	I/Q or API Data Select = 0
to_connector [33]	Word Identifier = 0	Word Identifier = 1
to_connector [32]	UNUSED	UNUSED
to_connector [31]	API Data [15]	UNUSED
to_connector [30]	API Data [14]	UNUSED
to_connector [29]	API Data [13]	UNUSED
to_connector [28]	API Data [12]	UNUSED
to_connector [27]	API Data [11]	UNUSED
to_connector [26]	API Data [10]	UNUSED
to_connector [25]	API Data [9]	UNUSED
to_connector [24]	API Data [8]	UNUSED
to_connector [23]	API Data [7]	API Data Identifier [7]
to_connector [22]	API Data [6]	API Data Identifier [6]
to_connector [21]	API Data [5]	API Data Identifier [5]
to_connector [20]	API Data [4]	API Data Identifier [4]
to_connector [19]	API Data [3]	API Data Identifier [3]
to_connector [18]	API Data [2]	API Data Identifier [2]
to_connector [17]	API Data [1]	API Data Identifier [1]
to_connector [16]	API Data [0]	API Data Identifier [0]
to_connector [15]	Running Time Stamp [15]	UNUSED
to_connector [14]	Running Time Stamp [14]	UNUSED
to_connector [13]	Running Time Stamp [13]	UNUSED
to_connector [12]	Running Time Stamp [12]	UNUSED
to_connector [11]	Running Time Stamp [11]	UNUSED
to_connector [10]	Running Time Stamp [10]	UNUSED
to_connector [9]	Running Time Stamp [9]	UNUSED
to_connector [8]	Running Time Stamp [8]	CI Number [8]
to_connector [7]	Running Time Stamp [7]	CI Number [7]
to_connector [6]	Running Time Stamp [6]	CI Number [6]
to_connector [5]	Running Time Stamp [5]	CI Number [5]
to_connector [4]	Running Time Stamp [4]	CI Number [4]
to_connector [3]	Running Time Stamp [3]	CI Number [3]
to_connector [2]	Running Time Stamp [2]	CI Number [2]
to_connector [1]	Running Time Stamp [1]	CI Number [1]
to_connector [0]	Running Time Stamp [0]	CI Number [0]

Table B.2: Antenna Processor Interface output data word format

BIBLIOGRAPHY

[1] Merrill I. Skolnik. *Radar Handbook*, chapter 17, pages 17.1–17.42. McGraw-Hill, New York, 2 edition, 1990.

[2] George W. Stimson. *Introduction to Airborne Radar.* Scitech, 2 edition, 1998.

[3] Merrill I. Skolnik. *Introduction to Radar Systems.* McGraw-Hill, New York, 3 edition, 1989.

[4] Dr. James J. Genova. Coherent seeker guided anti ship missile performance analysis, June 2004.

[5] Altera. Flex 10k embedded programmable logic device family data sheet, January 2003.

[6] Altera. Epf10k30 device pin-outs, 2001.

[7] Altera. Usb-blaster download cable user guide, July 2006.

[8] Altera. Altera configuration handbook, July 2004.

[9] Texas Instruments. Sn54lvth245a, sn74lvth245a 3.3-v abt octal bus transceivers with 3-state outputs, September 2003.

[10] Douglas Brooks. *Signal Integrity Issues and Printed Circuit Board Design.* Prentice Hall, 2003.

[11] Texas Instruments. Bus-interface devices with output-damping resistors or reduced-drive outputs, August 1997.

[12] Eric Bogatin. *Signal Integrity - Simplified.* Prentice Hall, 2003.